(12)特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2001 年11 月1 日 (01.11.2001)

PCT

(10) 国際公開番号
WO 01/82606 A1

(51) 国際特許分類[7]: H04N 5/92, G11B 20/10

(21) 国際出願番号: PCT/JP01/03415

(22) 国際出願日: 2001 年4 月20 日 (20.04.2001)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2000-183771 2000 年4 月21 日 (21.04.2000) JP
特願2000-271552 2000 年9 月7 日 (07.09.2000) JP

(71) 出願人 (米国を除く全ての指定国について): ソニー株式会社 (SONY CORPORATION) [JP/JP]; 〒141-0001 東京都品川区北品川6丁目7番35号 Tokyo (JP).

(72) 発明者; および
(75) 発明者/出願人 (米国についてのみ): 加藤元樹 (KATO, Motoki) [JP/JP]. 浜田俊也 (HAMADA, Toshiya) [JP/JP]; 〒141-0001 東京都品川区北品川6丁目7番35号 ソニー株式会社内 Tokyo (JP).

(74) 代理人: 小池 晃, 外(KOIKE, Akira et al.); 〒105-0001 東京都港区虎ノ門二丁目6番4号 第11森ビル Tokyo (JP).

(81) 指定国 (国内): AU, BR, CA, CN, CZ, HU, ID, IL, IN, KR, MX, NO, NZ, PL, RO, RU, SG, SK, US, VN, ZA.

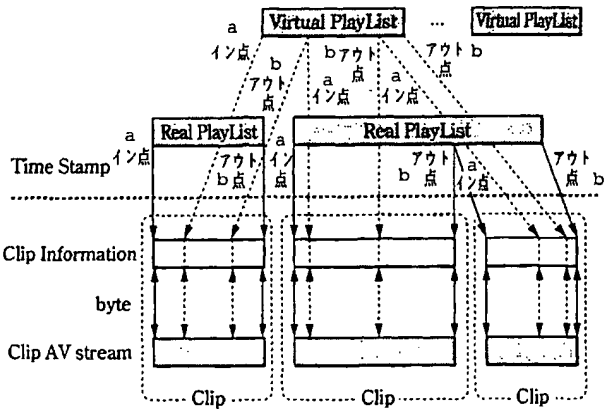
(84) 指定国 (広域): ヨーロッパ特許 (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

添付公開書類:
— 国際調査報告書
— 補正書

2文字コード及び他の略語については、定期発行される各PCTガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

(54) Title: INFORMATION PROCESSING APPARATUS AND METHOD, PROGRAM, AND RECORDED MEDIUM

(54) 発明の名称: 情報処理装置及び方法、プログラム、並びに記録媒体



(57) Abstract: A CPI-type is described in a PlayList(). The CPI-type includes an EP-map type and a TU-map type. If the position of an I picture can be analyzed, the EP-map type is used; and if not, the TU-map type is used. As a result, AV stream data recorded after analyzing the position of the I picture and AV stream data recorded without analyzing the position of the I picture can be commonly managed.

(57) 要約:

PlayList()には、CPI_typeが記述される。CPI_typeには、EP_maptypeと、TU_map typeがある。Ｉピクチャの位置が分析できる場合、EP_maptypeが用いられ、Ｉピクチャの位置が分析できない場合、TU_map typeが用いられる。これにより、Ｉピクチャの位置を分析してＡＶストリームデータと、分析しないで記録するＡＶストリームデータとを共通に管理できるようにする。

WO 01/82606 A1

明細書

情報処理装置及び方法、プログラム、並びに記録媒体

技術分野

本発明は情報処理装置、再生方法、プログラム、並びに記録媒体に関し、特に、ＧＵＩ等に説明表示する情報、主の再生経路の情報、副の再生経路の情報、主の再生経路を構成する個々の再生区間の間の接続情報、ユーザが所望したシーンにセットするブックマークやリジューム点の情報等の情報を含むファイルを記録する情報処理装置、再生方法、プログラム、並びに記録媒体に関する。

背景技術

近年、記録再生装置から取り外し可能なディスク型の記録媒体として、各種の光ディスクが提案されつつある。このような記録可能な光ディスクは、数ギガバイトの大容量メディアとして提案されており、ビデオ信号等のＡＶ(Audio Visual)信号を記録するメディアとしての期待が高い。この記録可能な光ディスクに記録するディジタルのＡＶ信号のソース（供給源）としては、ＣＳディジタル衛星放送やＢＳディジタル放送があり、また、将来はディジタル方式の地上波テレビジョン放送等も提案されている。

ここで、これらのソースから供給されるディジタルビデオ信号は、通常ＭＰＥＧ（Moving Picture Experts Group）２方式で画像圧縮されているのが一般的である。また、記録装置には、その装置固有の記録レートが定められている。従来の民生用映像蓄積メディアで、ディジタル放送由来のディジタルビデオ信号を記録する場合、アナログ記録方式であれば、ディジタルビデオ信号をデコード後、帯域制限をして記録する。或いは、ＭＰＥＧ１ Videio、ＭＰＥＧ２ Videio、ＤＶ方式をはじめとするディジタル記録方式であれば、１度デコードされた後に、その装置固有の記録レート・符号化方式で再エンコードされて記録される。

しかしながら、このような記録方法は、供給されたビットストリームを１度デコードし、その後で帯域制限や再エンコードを行って記録するため、画質の劣化を伴う。画像圧縮されたディジタル信号の記録をする場合、入力されたディジタル信号の伝送レートが記録再生装置の記録レートを超えない場合には、供給されたビットストリームをデコードや再エンコードすることなく、そのまま記録する方法が最も画質の劣化が少ない。但し、画像圧縮されたディジタル信号の伝送レートが記録媒体としてのディスクの記録レートを超える場合には、記録再生装置でデコード後、伝送レートがディスクの記録レートの上限以下になるように、再エンコードをして記録する必要はある。

また、入力ディジタル信号のビットレートが時間により増減する可変レート方式によって伝送されている場合には、回転ヘッドが固定回転数であるために記録レートが固定レートになるテープ記録方式に比べ、１度バッファにデータを蓄積し、バースト的に記録ができるディスク記録装置が記録媒体の容量をより無駄なく利用できる。

以上のように、ディジタル放送が主流となる将来においては、データストリーマのように放送信号をディジタル信号のまま、デコードや再エンコードすることなく記録し、記録媒体としてディスクを使用した記録再生装置が求められると予測される。

ところで、上述したような記録装置により記録媒体にＡＶストリームデータを記録する場合、例えば、高速再生ができるようにするために、Ａストリームデータを分析し、Ｉピクチャの位置を検出して、Ｉピクチャにアクセスできるようにして記録する場合と、ＡＶストリームデータを分析せず、そのまま記録する場合とがある。

このような場合、従来、それぞれ専用のアプリケーションプログラムを用意し、それぞれにより、ＡＶストリームを、異なるフォーマットのＡＶストリーム（高速再生が可能なＡＶストリーム、又は不可能なＡＶストリーム）として記録媒体に記録するようにしていた。この結果、アプリケーションプログラムの開発に、費用と時間がかかる課題があった。また、それぞれのアプリケーションプログラムにより記録されたＡＶストリームは、異なるフォーマットのものとなので、相

互の互換性がなくなり、共通の装置で再生することができなくなる課題があった。

更に、従来の記録装置では、例えば、オーディオデータを、所謂アフターレコーディングすることが困難である課題があった。

発明の開示

本発明の目的は、このような状況に鑑み、高速再生が可能なＡＶストリームと不可能なＡＶストリームを、共通に管理することにある。

また、本発明の他の目的は、アフターレコーディングを可能にすることにある。

本発明に係る情報処理装置は、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルを生成する第１の生成手段と、記録方法に応じて第１のテーブル又は第２のテーブルの一方を選択する選択手段と、選択されたテーブルをＡＶストリームデータとともに記録媒体に記録する第１の記録手段とを有する。

前記第１のテーブルは、EP_mapであり、第２のテーブルは、TU_mapとすることができる。

前記選択手段は、ノンコグニザント記録の際には、第２のテーブルを選択することができる。

前記選択手段は、セルフエンコード記録の際には、第１のテーブルを選択することができる。

前記選択手段は、コグニザント記録の際には、第１のテーブルを選択することができる。

前記ＡＶストリームデータの再生を指定する再生指定情報を生成する第２の生成手段と、第２の生成手段により生成された再生指定情報を記録媒体に記録する第２の記録手段を更に有し、再生指定情報は、ＡＶストリームデータの再生区間の時間情報を、プレゼンテーションタイムベースで表現するか、又はアライバル

タイムベースで表現するかを示す種別情報を含むようにすることができる。

前記ＡＶストリームデータとともに第１のテーブルが記録されている場合、再生指定情報は、ＡＶストリームデータの再生区間の時間情報を、プレゼンテーションタイムベースで表現し、ＡＶストリームデータとともに第２のテーブルが記録されている場合、再生指定情報は、ＡＶストリームデータの再生区間の時間情報を、アライバルタイムベースで表現することができる。

本発明に係る情報処理方法は、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルを生成する生成ステップと、記録方法に応じて第１のテーブル又は第２のテーブルの一方を選択する選択ステップと、選択されたテーブルをＡＶストリームデータとともに記録媒体に記録する記録ステップとを含む。

本発明に係る記録媒体のプログラムは、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルを生成する生成ステップと、記録方法に応じて第１のテーブル又は第２のテーブルの一方を選択する選択ステップと、選択されたテーブルをＡＶストリームデータと共に記録媒体に記録する記録ステップとを含む。

本発明に係るプログラムは、プレゼンテーションタイムスタンプと、これに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルを生成する生成ステップと、記録方法に応じて第１のテーブル又は第２のテーブルの一方を選択する選択ステップと、選択されたテーブルをＡＶストリームデータと共に記録

媒体に記録する記録ステップとをコンピュータに実行させる。

本発明に係る情報処理装置は、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルの一方が、記録方法に応じて記録されている記録媒体から、第１のテーブル又は第２のテーブルの一方を再生する再生手段と、再生されたテーブルに基づいて、ＡＶストリームデータの出力を制御する制御手段とを有する。

本発明に係る情報処理方法は、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルの一方が、記録方法に応じて記録されている記録媒体から、第１のテーブル又は第２のテーブルの一方を再生する再生ステップと、再生されたテーブルに基づいて、ＡＶストリームデータの出力を制御する制御ステップとを含む。

本発明に係る記録媒体のプログラムは、プレゼンテーションタイムスタンプと、これに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルの一方が、記録方法に応じて記録されている記録媒体から、第１のテーブル又は第２のテーブルの一方を再生する再生ステップと、再生されたテーブルに基づいて、ＡＶストリームデータの出力を制御する制御ステップとを含む。

本発明に係るプログラムは、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶスト

リームデータ中のアドレスとの対応関係を記述する第２のテーブルの一方が、記録方法に応じて記録されている記録媒体から、第１のテーブル又は第２のテーブルの一方を再生する再生ステップと、再生されたテーブルに基づいて、ＡＶストリームデータの出力を制御する制御ステップとをコンピュータに実行させる。

本発明に係る記録媒体は、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットのＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットのＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルの一方が、記録方法に応じて記録されている。

本発明に係る情報処理装置は、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を生成する生成手段と、ＡＶストリームデータと再生指定情報を記録媒体に記録する記録手段とを備える。

前記副の再生パスは、オーディオデータのアフターレコーディング用のパスとすることができる。

前記第１の情報は、Main_pathであり、第２の情報は、Sub_pathとすることができる。

前記第２の情報は、副の再生パスのタイプを表すタイプ情報、副の再生パスが参照するＡＶストリームのファイル名、副の再生パスのＡＶストリームのイン点とアウト点、及び再生パスのイン点が、主のパスの時間軸上で同期してスタートする主のパス上の時刻を含むようにすることができる。

本発明に係る情報処理方法は、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を生成する生成ステップと、ＡＶストリームデータと再生指定情報を記録媒体に記録する記録ステップとを含む。

本発明に係る記録媒体のプログラムは、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を生成する生成ステップと、ＡＶストリームデータと再生指定情

報を記録媒体に記録する記録ステップとを含む。

本発明に係るプログラムは、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を生成する生成ステップと、ＡＶストリームデータと再生指定情報を記録媒体に記録する記録ステップとをコンピュータに実行させる。

前記情報処理装置は、主の再生パスを示す第１の情報と、前記主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を、前記記録媒体から再生する再生手段と、再生された前記再生指定情報に基づいて、前記ＡＶストリームデータの出力を制御する制御手段とを備える。

本発明に係る情報処理方法は、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を、記録媒体から再生する再生ステップと、再生された再生指定情報に基づいて、ＡＶストリームデータの出力を制御する制御ステップとを含む。

本発明に係る記録媒体のプログラムは、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を、記録媒体から再生する再生ステップと、再生された再生指定情報に基づいて、ＡＶストリームデータの出力を制御する制御ステップとを含む。

本発明に係るプログラムは、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報を、記録媒体から再生する再生ステップと、再生された再生指定情報に基づいて、ＡＶストリームデータの出力を制御する制御ステップとをコンピュータに実行させる。

本発明に係る記録媒体は、主の再生パスを示す第１の情報と、主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報が記録されている。

本発明に係る情報処理装置及び方法、記録媒体のプログラム、プログラム、並びに記録媒体においては、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットの前記ＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたア

ライバルタイムスタンプと、それに対応するトランスポートパケットの前記ＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルの一方が、記録方法に応じて記録される。

本発明に係る情報処理装置及び方法、記録媒体のプログラム、並びにプログラムにおいては、プレゼンテーションタイムスタンプと、それに対応するアクセスユニットの前記ＡＶストリームデータ中のアドレスとの対応関係を記述する第１のテーブル、又は、トランスポートパケットの到着時刻に基づいたアライバルタイムスタンプと、それに対応するトランスポートパケットの前記ＡＶストリームデータ中のアドレスとの対応関係を記述する第２のテーブルの一方が、記録方法に応じて記録されている記録媒体から、そのテーブルが再生され、それに基づいて、出力が制御される。

本発明に係る情報処理装置及び方法、記録媒体のプログラム、プログラム、並びに第２の記録媒体においては、主の再生パスを示す第１の情報と、前記主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報が記録される。

本発明に係る情報処理装置及び方法、記録媒体のプログラム、並びにプログラムにおいては、主の再生パスを示す第１の情報と、前記主の再生パスと同期して再生される副の再生パスを示す第２の情報により構成される再生指定情報が記録媒体から再生され、それに基づいて、出力が制御される。

本発明の更に他の目的、特徴や利点は、後述する本発明の実施例や添付する図面に基づくより詳細な説明によって明らかになるであろう。

図面の簡単な説明

図１は、本発明を適用した記録再生装置の構成を示す図である。

図２は、記録再生装置により記録媒体に記録されるデータのフォーマットについて説明する図である。

図３は、Real PlayListとVirtual PlayListについて説明する図である。

図４Ａ～図４Ｃは、Real PlayListの作成について説明する図である。

図５Ａ～図５Ｃは、Real PlayListの削除について説明する図である。

図６Ａ及び図６Ｂは、アセンブル編集について説明する図である。

図７は、Virtual PlayListにサブパスを設ける場合について説明する図である。

図８は、PlayListの再生順序の変更について説明する図である。

図９は、PlayList上のマークとClip上のマークについて説明する図である。

図１０は、メニューサムネイルについて説明する図である。

図１１は、PlayListに付加されるマークについて説明する図である。

図１２は、クリップに付加されるマークについて説明する図である。

図１３は、PlayList、Clip、サムネイルファイルの関係について説明する図である。

図１４は、ディレクトリ構造について説明する図である。

図１５は、info.dvrのシンタクスを示す図である。

図１６は、DVR volumeのシンタクスを示す図である。

図１７は、Resumevolumeのシンタクスを示す図である。

図１８は、UIAppInfovolumeのシンタクスを示す図である。

図１９は、Character set valueのテーブルを示す図である。

図２０は、TableOfPlayListのシンタクスを示す図である。

図２１は、TableOfPlayListの他のシンタクスを示す図である。

図２２は、MakersPrivateDataのシンタクスを示す図である。

図２３は、xxxxx.rplsとyyyyy.vplsのシンタクスを示す図である。

図２４Ａ～図２４Ｃは、PlayListについて説明する図である。

図２５は、PlayListのシンタクスを示す図である。

図２６は、PlayList_typeのテーブルを示す図である。

図２７は、UIAppinfoPlayListのシンタクスを示す図である。

図２８Ａ～図２８Ｃは、図２７に示したUIAppinfoPlayListのシンタクス内のフラグについて説明する図である。

図２９は、PlayItemについて説明する図である。

図３０は、PlayItemについて説明する図である。

図３１は、PlayItemについて説明する図である。

図３２は、PlayItemのシンタクスを示す図である。

図３３は、IN_timeについて説明する図である。

図３４は、OUT_timeについて説明する図である。

図３５は、Connection_Conditionのテーブルを示す図である。

図３６Ａ～図３６Ｄは、Connection_Conditionについて説明する図である。

図３７は、BridgeSequenceInfoを説明する図である。

図３８は、BridgeSequenceInfoのシンタクスを示す図である。

図３９は、SubPlayItemについて説明する図である。

図４０は、SubPlayItemのシンタクスを示す図である。

図４１は、SubPath_typeのテーブルを示す図である。

図４２は、PlayListMarkのシンタクスを示す図である。

図４３は、Mark_typeのテーブルを示す図である。

図４４は、Mark_time_stampを説明する図である。

図４５は、zzzzz.clipのシンタクスを示す図である。

図４６は、ClipInfoのシンタクスを示す図である。

図４７は、Clip_stream_typeのテーブルを示す図である。

図４８は、offset_SPNについて説明する図である。

図４９は、offset_SPNについて説明する図である。

図５０Ａ及び図５０Ｂは、ＳＴＣ区間について説明する図である。

図５１は、STC_Infoについて説明する図である。

図５２は、STC_Infoのシンタクスを示す図である。

図５３は、ProgramInfoを説明する図である。

図５４は、ProgramInfoのシンタクスを示す図である。

図５５は、VideoCondingInfoのシンタクスを示す図である。

図５６は、Video_formatのテーブルを示す図である。

図５７は、frame_rateのテーブルを示す図である。

図５８は、display_aspect_ratioのテーブルを示す図である。

図５９は、AudioCondingInfoのシンタクスを示す図である。

図６０は、audio_codingのテーブルを示す図である。

図６１は、audio_component_typeのテーブルを示す図である。

図６２は、sampling_frequencyのテーブルを示す図である。

図６３は、CPIについて説明する図である。

図６４は、CPIについて説明する図である。

図６５は、CPIのシンタクスを示す図である。

図６６は、CPI_typeのテーブルを示す図である。

図６７は、ビデオEP_mapについて説明する図である。

図６８は、EP_mapについて説明する図である。

図６９は、EP_mapについて説明する図である。

図７０は、EP_mapのシンタクスを示す図である。

図７１は、EP_type valuesのテーブルを示す図である。

図７２は、EP_map_for_one_stream_PIDのシンタクスを示す図である。

図７３は、TU_mapについて説明する図である。

図７４は、TU_mapのシンタクスを示す図である。

図７５は、ClipMarkのシンタクスを示す図である。

図７６は、mark_typeのテーブルを示す図である。

図７７は、mark_type_stampのテーブルを示す図である。

図７８は、menu.thmbとmark.thmbのシンタクスを示す図である。

図７９は、Thumbnailのシンタクスを示す図である。

図８０は、thumbnail_picture_formatのテーブルを示す図である。

図８１Ａ及び図８１Ｂは、tn_blockについて説明する図である。

図８２は、DVR MPEG２のトランスポートストリームの構造について説明する図である。

図８３は、DVR MPEG２のトランスポートストリームのレコーダモデルを示す図である。

図８４は、DVR MPEG２のトランスポートストリームのプレーヤモデルを示す図である。

図８５は、source packetのシンタクスを示す図である。

図８６は、TP_extra_headerのシンタクスを示す図である。

図８７は、copy permission indicatorのテーブルを示す図である。

図８８は、シームレス接続について説明する図である。

図８９は、シームレス接続について説明する図である。

図９０は、シームレス接続について説明する図である

図９１は、シームレス接続について説明する図である。

図９２は、シームレス接続について説明する図である

図９３は、オーディオのオーバーラップについて説明する図である。

図９４は、BridgeSequenceを用いたシームレス接続について説明する図である。

図９５は、BridgeSequenceを用いないシームレス接続について説明する図である。

図９６は、DVR STDモデルを示す図である。

図９７は、復号、表示のタイミングチャートである。

図９８は、PlayListファイルのシンタクスを示す図である。

図９９は、図９８のPlayListファイル中のUIAppInfoPlayListのシンタクスを示す図である。

図１００は、図９８のPlayListファイル中のPlayList()のシンタクスを示す図である。

図１０１は、SubPlayItemのシンタクスを示す図である。

図１０２は、Real PlayListの作成方法を説明するフローチャートである。

図１０３は、Virtual PlayListの作成方法を説明するフローチャートである。

図１０４は、PlayListの再生方法を説明するフローチャートである。

図１０５は、PlayListのSubパスの再生方法を説明するフローチャートである。

図１０６は、PlayListMarkの作成方法を説明するフローチャートである。

図１０７は、PlayListMarkを使用した頭出し再生方法を説明するフローチャートである。

図１０８は、媒体を説明する図である。

発明を実施するための最良の形態

以下に、本発明が適用された情報処理装置及び方法、プログラム、並びに記録媒体について、図面を参照して説明する。図１は、本発明を適用した記録再生装置１の内部構成例を示す図である。先ず、外部から入力された信号を記録媒体に記録する動作を行う部分の構成について説明する。記録再生装置１は、アナログデータ、又は、ディジタルデータを入力し、記録することができる構成とされている。

端子１１には、アナログのビデオ信号が、端子１２には、アナログのオーディオ信号が、それぞれ入力される。端子１１に入力されたビデオ信号は、解析部１４とＡＶエンコーダ１５に、それぞれ出力される。端子１２に入力されたオーディオ信号は、ＡＶエンコーダ１５に出力される。解析部１４は、入力されたビデオ信号からシーンチェンジ等の特徴点を抽出する。

ＡＶエンコーダ１５は、入力されたビデオ信号とオーディオ信号を、それぞれ符号化し、符号化ビデオストリーム（Ｖ）、符号化オーディオストリーム（Ａ）、及びＡＶ同期等のシステム情報（Ｓ）をマルチプレクサ１６に出力する。

符号化ビデオストリームは、例えば、ＭＰＥＧ（Moving Picture Expert Group）２方式により符号化されたビデオストリームであり、符号化オーディオストリームは、例えば、ＭＰＥＧ１方式により符号化されたオーディオストリームや、ドルビーＡＣ３方式により符号化されたオーディオストリーム等である。マルチプレクサ１６は、入力されたビデオ及びオーディオのストリームを、入力システム情報に基づいて多重化して、スイッチ１７を介して多重化ストリーム解析部１８とソースパケッタイザ１９に出力する。

多重化ストリームは、例えば、ＭＰＥＧ２トランスポートストリームやＭＰＥＧ２プログラムストリームである。ソースパケッタイザ１９は、入力された多重化ストリームを、このストリームを記録させる記録媒体１００のアプリケーションフォーマットに従って、ソースパケットから構成されるＡＶストリームを符号化する。ＡＶストリームは、ＥＣＣ（誤り訂正）符号化部２０、変調部２１で所定の処理が施され、書込部２２に出力される。書込部２２は、制御部２３から出力される制御信号に基づいて、記録媒体１００にＡＶストリームファイルを書き込む（記録する）。

ディジタルインタフェース又はディジタルテレビジョンチューナから入力されるディジタルテレビジョン放送等のトランスポートストリームは、端子１３に入力される。端子１３に入力されたトランスポートストリームの記録方式には、２通りあり、これらは、トランスペアレントに記録する方式と、記録ビットレートを下げる等の目的のために再エンコードをした後に記録する方式である。記録方式の指示情報は、ユーザインタフェースとしての端子２４から制御部２３へ入力される。

入力トランスポートストリームをトランスペアレントに記録する場合、端子１３に入力されたトランスポートストリームは、多重化ストリーム解析部１８と、ソースパケッタイザ１９に出力される。これ以降の記録媒体１００へＡＶストリームが記録されるまでの処理は、上述の入力オーディオ浸透とビデオ信号を符号化して記録する場合と同一の処理なので、その説明は省略する。

入力トランスポートストリームを再エンコードした後に記録する場合、端子１３に入力されたトランスポートストリームは、デマルチプレクサ２６に入力される。デマルチプレクサ２６は、入力されたトランスポートストリームに対してデマルチプレクス処理を施し、ビデオストリーム（Ｖ）、オーディオストリーム（Ａ）、及びシステム情報（Ｓ）を抽出する。

デマルチプレクサ２６により抽出されたストリーム（情報）のうち、ビデオストリームはＡＶデコーダ２７に、オーディオストリームとシステム情報はマルチプレクサ１６に、それぞれ出力される。ＡＶデコーダ２７は、入力されたビデオストリームを復号し、その再生ビデオ信号をＡＶエンコーダ１５に出力する。ＡＶエンコーダ１５は、入力ビデオ信号を符号化し、符号化ビデオストリーム（Ｖ）をマルチプレクサ１６に出力する。

一方、デマルチプレクサ２６から出力され、マルチプレクサ１６に入力されたオーディオストリームとシステム情報、及び、ＡＶエンコーダ１５から出力されたビデオストリームは、入力システム情報に基づいて、多重化されて、多重化ストリームとして多重化ストリーム解析部１８とソースパケットタイザ１９にスイッチ１７を介して出力される。これ以後の記録媒体１００へＡＶストリームが記録されるまでの処理は、上述の入力オーディオ信号とビデオ信号を符号化して記

録する場合と同一の処理なので、その説明は省略する。

記録再生装置１は、ＡＶストリームのファイルを記録媒体１００に記録すると共に、そのファイルを説明するアプリケーションデータベース情報も記録する。アプリケーションデータベース情報は、制御部２３により作成される。制御部２３への入力情報は、解析部１４からの動画像の特徴情報、多重化ストリーム解析部１８からのＡＶストリームの特徴情報、及び端子２４から入力されるユーザからの指示情報である。

解析部１４から供給される動画像の特徴情報は、入力動画像信号の中の特徴的な画像に関係する情報であり、例えば、プログラムの開始点、シーンチェンジ点、コマーシャル（ＣＭ）の開始・終了点等の指定情報（マーク）であり、また、その指定場所の画像のサムネイル画像の情報も含まれる。

多重化ストリーム解析部１８からのＡＶストリームの特徴情報は、記録されるＡＶストリームの符号化情報に関係する情報であり、例えば、ＡＶストリーム内のＩピクチャのアドレス情報、ＡＶストリームの符号化パラメータ、ＡＶストリームの中の符号化パラメータの変化点情報、ビデオストリームの中の特徴的な画像に関係する情報（マーク）等である。

端子２４からのユーザの指示情報は、ＡＶストリームの中の、ユーザが指定した再生区間の指定情報、その再生区間の内容を説明するキャラクター文字、ユーザが好みのシーンにセットするブックマークやリジューム点の情報等である。

制御部２３は、上記の入力情報に基づいて、ＡＶストリームのデータベース(Clip)、ＡＶストリームの再生区間(PlayItem)をグループ化したもの（PlayList)のデータベース、記録媒体１００の記録内容の管理情報(info.dvr)、及びサムネイル画像の情報を作成する。これらの情報から構成されるアプリケーションデータベース情報は、ＡＶストリームと同様にして、ＥＣＣ符号化部２０、変調部２１で処理されて、書込部２２へ入力される。書込部２２は、制御部２３から出力される制御信号に基づいて、記録媒体１００へデータベースファイルを記録する。

上述したアプリケーションデータベース情報についての詳細は後述する。

このようにして記録媒体１００に記録されたＡＶストリームファイル（画像データと音声データのファイル）と、アプリケーションデータベース情報が再生さ

れる場合、先ず、制御部２３は、読出部２８に対して、記録媒体１００からアプリケーションデータベース情報を読み出すように指示する。そして、読出部２８は、記録媒体１００からアプリケーションデータベース情報を読み出し、そのアプリケーションデータベース情報は、復調部２９、ＥＣＣ復号部３０の処理を経て、制御部２３へ入力される。

制御部２３は、アプリケーションデータベース情報に基づいて、記録媒体１００に記録されているPlayListの一覧を端子２４のユーザインタフェースへ出力する。ユーザは、PlayListの一覧から再生したいPlayListを選択し、再生を指定されたPlayListに関する情報が制御部２３へ入力される。制御部２３は、そのPlayListの再生に必要なＡＶストリームファイルの読み出しを、読出部２８に指示する。読出部２８は、その指示に従い、記録媒体１００から対応するＡＶストリームを読み出し復調部２９に出力する。復調部２９に入力されたＡＶストリームは、所定の処理が施されることにより復調され、更にＥＣＣ復号部３０の処理を経て、ソースデパケッタイザ３１出力される。

ソースデパケッタイザ３１は、記録媒体１００から読み出され、所定の処理が施されたアプリケーションフォーマットのＡＶストリームを、デマルチプレクサ２６に出力できるストリームに変換する。デマルチプレクサ２６は、制御部２３により指定されたＡＶストリームの再生区間(PlayItem)を構成するビデオストリーム（Ｖ）、オーディオストリーム（Ａ）、及びＡＶ同期等のシステム情報（Ｓ）を、ＡＶデコーダ２７に出力する。ＡＶデコーダ２７は、ビデオストリームとオーディオストリームを復号し、再生ビデオ信号と再生オーディオ信号を、それぞれ対応する端子３２と端子３３から出力する。

また、ユーザインタフェースとしての端子２４から、ランダムアクセス再生や特殊再生を指示する情報が入力された場合、制御部２３は、ＡＶストリームのデータベース(Clip)の内容に基づいて、記憶媒体１００からのＡＶストリームの読出位置を決定し、そのＡＶストリームの読み出しを、読出部２８に指示する。例えば、ユーザにより選択されたPlayListを、所定の時刻から再生する場合、制御部２３は、指定された時刻に最も近いタイムスタンプを持つＩピクチャからのデータを読み出すように読出部２８に指示する。

また、ユーザによって高速再生(Fast-forward playback)が指示された場合、制御部２３は、ＡＶストリームのデータベース(Clip)に基づいて、ＡＶストリームの中のI-ピクチャデータを順次連続して読み出すように読出部２８に指示する。

読出部２８は、指定されたランダムアクセスポイントからＡＶストリームのデータを読み出し、読み出されたデータは、後段の各部の処理を経て再生される。

次に、ユーザが、記録媒体１００に記録されているＡＶストリームの編集をする場合を説明する。ユーザが、記録媒体１００に記録されているＡＶストリームの再生区間を指定して新しい再生経路を作成したい場合、例えば、番組Ａという歌番組から歌手Ａの部分を再生し、その後続けて、番組Ｂという歌番組の歌手Ａの部分を再生したいといった再生経路を作成したい場合、ユーザインタフェースとしての端子２４から再生区間の開始点（イン点）と終了点（アウト点）の情報が制御部２３に入力される。制御部２３は、ＡＶストリームの再生区間(PlayItem)をグループ化したもの（PlayList）のデータベースを作成する。

ユーザが、記録媒体１００に記録されているＡＶストリームの一部を消去したい場合、ユーザインタフェースとしての端子２４から消去区間のイン点とアウト点の情報が制御部２３に入力される。制御部２３は、必要なＡＶストリーム部分だけを参照するようにPlayListのデータベースを変更する。また、ＡＶストリームの不必要なストリーム部分を消去するように、書込部２２に指示する。

ユーザが、記録媒体１００に記録されているＡＶストリームの再生区間を指定して新しい再生経路を作成したい場合であり、且つ、それぞれの再生区間をシームレスに接続したい場合について説明する。このような場合、制御部２３は、ＡＶストリームの再生区間(PlayItem)をグループ化したもの（PlayList）のデータベースを作成し、更に、再生区間の接続点付近のビデオストリームの部分的な再エンコードと再多重化を行う。

先ず、端子２４から再生区間のイン点のピクチャの情報と、アウト点のピクチャの情報が制御部２３へ入力される。制御部２３は、読出部２８にイン点側ピクチャとアウト点側のピクチャを再生するために必要なデータの読み出しを指示する。そして、読出部２８は、記録媒体１００からデータを読み出し、そのデータは、復調部２９、ＥＣＣ復号部３０、ソースデパケッタイザ３１を経て、デマル

チプレクサ２６に出力される。

制御部２３は、デマルチプレクサ２６に入力されたデータを解析して、ビデオストリームの再エンコード方法（picture_coding_typeの変更、再エンコードする符号化ビット量の割り当て）と、再多重化方式を決定し、その方式をＡＶエンコーダ１５とマルチプレクサ１６に供給する。

次に、デマルチプレクサ２６は、入力されたストリームをビデオストリーム（Ｖ）、オーディオストリーム（Ａ）、及びシステム情報（Ｓ）に分離する。ビデオストリームは、「ＡＶデコーダ２７に入力されるデータ」と「マルチプレクサ１６に入力されるデータ」がある。前者のデータは、再エンコードするために必要なデータであり、これはＡＶデコーダ２７で復号され、復号されたピクチャはＡＶエンコーダ１５で再エンコードされて、ビデオストリームにされる。後者のデータは、再エンコードをしないで、オリジナルのストリームからコピーされるデータである。オーディオストリーム、システム情報については、直接、マルチプレクサ１６に入力される。

マルチプレクサ１６は、制御部２３から入力された情報に基づいて、入力ストリームを多重化し、多重化ストリームを出力する。多重化ストリームは、ＥＣＣ符号化部２０、変調部２１で処理されて、書込部２２に入力される。書込部２２は、制御部２３から供給される制御信号に基づいて、記録媒体１００にＡＶストリームを記録する。

以下、アプリケーションデータベース情報や、その情報に基づく再生、編集といった操作に関する説明をする。図２は、アプリケーションフォーマットの構造を説明する図である。アプリケーションフォーマットは、ＡＶストリームの管理のためにPlayListとClipの２つのレイヤをもつ。Volume Informationは、ディスク内の全てのClipとPlayListの管理をする。ここでは、１つのＡＶストリームとその付属情報のペアを１つのオブジェクトとし、これをClipという。ＡＶストリームファイルはClip AV stream fileといい、この付属情報は、Clip Information fileという。

１つのClip AV stream fileは、ＭＰＥＧ２トランスポートストリームをアプリケーションフォーマットによって規定される構造に配置したデータをストアする。

一般的に、ファイルは、バイト列として扱われるが、Clip AV stream fileのコンテンツは、時間軸上に展開され、Clipの中のエントリポイントは、主に時間ベースで指定される。所定のClipへのアクセスポイントのタイムスタンプが与えられたとき、Clip Information fileは、Clip AV stream fileの中でデータの読み出しを開始すべきアドレス情報を見つけるために役立つ。

PlayListについて、図３を参照して説明する。PlayListは、Clipの中からユーザが見たい再生区間を選択し、これを簡単に編集することができるようにするために設けられている。１つのPlayListは、Clipの中の再生区間の集まりである。所定のClipの中の１つの再生区間は、PlayItemと呼ばれ、それは、時間軸上のイン点（ＩＮ）とアウト点（ＯＵＴ）の対で表される。したがって、PlayListは、複数のPlayItemが集まることにより構成される。

PlayListには、２つのタイプがある。１つは、Real PlayListであり、もう１つは、Virtual PlayListである。Real PlayListは、それが参照しているClipのストリーム部分を共有している。すなわち、Real PlayListは、それの参照しているClipのストリーム部分に相当するデータ容量をディスクの中で占め、Real PlayListが消去された場合、それが参照しているClipのストリーム部分もまたデータが消去される。

Virtual PlayListは、Clipのデータを共有していない。したがって、Virtual PlayListが変更又は消去されたとしても、Clipの内容には何も変化が生じない。

次に、Real PlayListの編集について説明する。図４Ａは、Real PlayListのクリエイト(create：作成)に関する図であり、ＡＶストリームが新しいClipとして記録される場合、そのClip全体を参照するReal PlayListが新たに作成される操作である。

図４Ｂは、Real PlayListのディバイド(divide：分割)に関する図であり、Real PlayListが所望な点で分けられて、２つのReal PlayListに分割される操作である。この分割という操作は、例えば、１つのPlayListにより管理される１つのクリップ内に、２つの番組が管理されているような場合に、ユーザが１つ１つの番組として登録（記録）し直したいといったようなときに行われる。この操作により、Clipの内容が変更される（Clip自体が分割される）ことはない。

図４Ｃは、Real PlayListのコンバイン(combine：結合)に関する図であり、２つのReal PlayListを結合して、１つの新しいReal PlayListにする操作である。この結合という操作は、例えば、ユーザが２つの番組を１つの番組として登録し直したいといったようなときに行われる。この操作により、Clipが変更される（Clip自体が１つにされる）ことはない。

図５Ａは、Real PlayList全体のデリート(delete：削除)に関する図であり、所定のReal PlayList全体を消去する操作がされた場合、削除されたReal PlayListが参照するClipの、対応するストリーム部分も削除される。

図５Ｂは、Real PlayListの部分的な削除に関する図であり、Real PlayListの所望な部分が削除された場合、対応するPlayItemが、必要なClipのストリーム部分だけを参照するように変更される。そして、Clipの対応するストリーム部分は削除される。

図５Ｃは、Real PlayListのミニマイズ(Minimize：最小化)に関する図であり、Real PlayListに対応するPlayItemを、Virtual PlayListに必要なClipのストリーム部分だけを参照するようにする操作である。Virtual PlayList にとって不必要なClipの、対応するストリーム部分は削除される。

上述したような操作により、Real PlayListが変更されて、そのReal PlayListが参照するClipのストリーム部分が削除された場合、その削除されたClipを使用しているVirtual PlayListが存在し、そのVirtual PlayListにおいて、削除されたClipにより問題が生じる可能性がある。

そのようなことが生じないように、ユーザに、削除という操作に対して、「そのReal PlayListが参照しているClipのストリーム部分を参照しているVirtual PlayListが存在し、もし、そのReal PlayListが消去されると、そのVirtual PlayListもまた消去されることになるが、それでも良いか？」といったメッセージ等を表示させることにより、確認（警告）を促した後に、ユーザの指示により削除の処理を実行、又は、キャンセルする。又は、Virtual PlayListを削除する代わりに、Real PlayListに対してミニマイズの操作が行われるようにする。

次にVirtual PlayListに対する操作について説明する。Virtual PlayListに対して操作が行われたとしても、Clipの内容が変更されることはない。図６Ａ及び

図６Ｂは、アセンブル(Assemble) 編集 (IN-OUT 編集)に関する図であり、ユーザが見たいと
所望した再生区間のPlayItemを作り、Virtual PlayListを作成するといった操作である。PlayItem間のシームレス接続が、アプリケーションフォーマットによりサポートされている（後述）。

図６Ａに示したように、２つのReal PlayList１，２と、それぞれのReal PlayListに対応するClip１，２が存在している場合に、ユーザがReal PlayList１内の所定の区間（In１乃至Out１までの区間：PlayItem１）を再生区間として指示し、続けて再生する区間として、Real PlayList２内の所定の区間（Ｉｎ２乃至Ｏｕｔ２までの区間：PlayItem２）を再生区間として指示したとき、図６Ｂに示すように、PlayItem１とPlayItem２から構成される１つのVirtual PlayListが作成される。

次に、Virtual PlayList の再編集(Re-editing)について説明する。再編集には、Virtual PlayListの中のイン点やアウト点の変更、Virtual PlayListへの新しいPlayItemの挿入(insert)や追加(append)、Virtual PlayListの中のPlayItemの削除等がある。また、Virtual PlayListそのものを削除することもできる。

図７は、Virtual PlayListへのオーディオのアフレコ(Audio dubbing (post recording))に関する図であり、Virtual PlayListへのオーディオのアフレコをサブパスとして登録する操作のことである。このオーディオのアフレコは、アプリケーションフォーマットによりサポートされている。Virtual PlayListのメインパスのＡＶストリームに、付加的なオーディオストリームが、サブパスとして付加される。

Real PlayListとVirtual PlayListで共通の操作として、図８に示すようなPlayListの再生順序の変更(Moving)がある。この操作は、ディスク(ボリューム)の中でのPlayListの再生順序の変更であり、アプリケーションフォーマットにおいて定義されるTable Of PlayList（図２０等を参照して後述する）によってサポートされる。この操作により、Clipの内容が変更されるようなことはない。

次に、マーク（Mark）について説明する。マークは、Clip及びPlayListの中のハイライトや特徴的な時間を指定するために設けられている。Clipに付加される

マークは、ＡＶストリームの内容に起因する特徴的なシーンを指定する、例えば、シーンチェンジ点等である。PlayListを再生するとき、そのPlayListが参照するClipのマークを参照して、使用することができる。

PlayListに付加されるマークは、主にユーザによってセットされる、例えば、ブックマークやリジューム点等である。Clip又はPlayListにマークをセットすることは、マークの時刻を示すタイムスタンプをマークリストに追加することにより行われる。また、マークを削除することは、マークリストの中から、そのマークのタイムスタンプを除去することである。したがって、マークの設定や削除により、ＡＶストリームは何の変更もされない。

次に、サムネイルについて説明する。サムネイルは、Volume、PlayList、及びClipに付加される静止画である。サムネイルには、２つの種類があり、１つは、内容を表す代表画としてのサムネイルである。これは主としてユーザがカーソル（不図示）等を操作して見たいものを選択するためのメニュー画面で使われるものである。もう１つは、マークが指しているシーンを表す画像である。

Volumeと各Playlistは代表画を持つことができるようにする必要がある。Volumeの代表画は、ディスク（記録媒体１００、以下、記録媒体１００はディスク状のものであるとし、適宜、ディスクと記述する）を記録再生装置１の所定の場所にセットしたときに、そのディスクの内容を表す静止画を最初に表示する場合等に用いられることを想定している。Playlistの代表画は、Playlistを選択するメニュー画面において、Playlistの内容を表すための静止画として用いられることを想定している。

Playlistの代表画として、Playlistの最初の画像をサムネイル（代表画）にすることが考えられるが、必ずしも再生時刻０の先頭の画像が内容を表す上で最適な画像とは限らない。そこで、Playlistのサムネイルとして、任意の画像をユーザが設定できるようにする。以上２種類のサムネイルをメニューサムネイルという。メニューサムネイルは頻繁に表示されるため、ディスクから高速に読み出される必要がある。このため、全てのメニューサムネイルを１つのファイルに格納することが効率的である。メニューサムネイルは、必ずしもボリューム内の動画から抜き出したピクチャである必要はなく、図１０に示すように、パーソナルコ

ンピュータやディジタルスチルカメラから取り込まれた画像でもよい。

一方、ClipとPlaylistには、複数個のマークを打てる必要があり、マーク位置の内容を知るためにマーク点の画像を容易に見ることができるようにする必要がある。このようなマーク点を表すピクチャをマークサムネイル（Mark Thumbnails）という。したがって、サムネイルの元となる画像は、外部から取り込んだ画像よりも、マーク点の画像を抜き出したものが主となる。

図１１は、PlayListに付けられるマークと、そのマークサムネイルの関係について示す図であり、図１２は、Clipに付けられるマークと、そのマークサムネイルの関係について示す図である。マークサムネイルは、メニューサムネイルと異なり、Playlistの詳細を表すときに、サブメニュー等で使われるため、短いアクセス時間で読み出されるようなことは要求されない。そのため、サムネイルが必要になる度に、記録再生装置１がファイルを開き、そのファイルの一部を読み出すことで多少時間がかかっても、問題にはならない。

また、ボリューム内に存在するファイル数を減らすために、全てのマークサムネイルは１つのファイルに格納するのがよい。Playlistはメニューサムネイル１つと複数のマークサムネイルを有することができるが、Clipは直接ユーザが選択する必要性がない（通常、Playlist経由で指定する）ため、メニューサムネイルを設ける必要はない。

図１３は、上述したことを考慮した場合のメニューサムネイル、マークサムネイル、PlayList、及びClipの関係について示した図である。メニューサムネイルファイルには、PlayList毎に設けられたメニューサムネイルがファイルされている。メニューサムネイルファイルには、ディスクに記録されているデータの内容を代表するボリュームサムネイルが含まれている。マークサムネイルファイルは、各PlayList毎と各Clip毎に作成されたサムネイルがファイルされている。

次に、ＣＰＩ（Characteristic Point Information）について説明する。ＣＰＩは、Clipインフォメーションファイルに含まれるデータであり、主に、それはClipへのアクセスポイントのタイムスタンプが与えられたとき、Clip AV stream fileの中でデータの読み出しを開始すべきデータアドレスを見つけるために用いられる。本実施の形態では、２種類のＣＰＩを用いる。１つは、EP_mapであり、

もう１つは、TU_mapである。

EP_mapは、エントリポイント（ＥＰ）データのリストであり、これは、エレメンタリーストリーム及びトランスポートストリームから抽出されたものである。これは、ＡＶストリームの中でデコードを開始すべきエントリポイントの場所を見つけるためのアドレス情報を持つ。１つのＥＰデータは、プレゼンテーションタイムスタンプ（ＰＴＳ）と、そのＰＴＳに対応するアクセスユニットのＡＶストリームの中のデータアドレスの対で構成される。

EP_mapは、主に２つの目的のために使用される。第１に、PlayListの中でプレゼンテーションタイムスタンプによって参照されるアクセスユニットのＡＶストリームの中のデータアドレスを見つけるために使用される。第２に、ファーストフォワード再生やファーストリバース再生のために使用される。記録再生装置１が、入力ＡＶストリームを記録する場合、そのストリームのシンタクスを解析することができるとき、EP_mapが作成され、ディスクに記録される。

TU_mapは、ディジタルインタフェースを通して入力されるトランスポートパケットの到着時刻に基づいたタイムユニット（ＴＵ）データのリストを持つ。これは、到着時刻ベースの時間とＡＶストリームの中のデータアドレスとの関係を与える。記録再生装置１が、入力ＡＶストリームを記録する場合、そのストリームのシンタクスを解析することができないとき、TU_mapが作成され、ディスクに記録される。

STCInfoは、ＭＰＥＧ２トランスポートストリームをストアしているＡＶストリームファイルの中にあるＳＴＣの不連続点情報をストアする。ＡＶストリームがＳＴＣの不連続点を持つ場合、そのＡＶストリームファイルの中で同じ値のＰＴＳが現れるかもしれない。そのため、ＡＶストリーム上のある時刻をＰＴＳベースで指す場合、アクセスポイントのＰＴＳだけではそのポイントを特定するためには不十分である。更に、そのＰＴＳを含むところの連続なＳＴＣ区間のインデックスが必要である。連続なＳＴＣ区間を、このフォーマットでは STC-sequenceと呼び、そのインデックスをSTC-sequence-idと呼ぶ。STC-sequenceの情報は、Clip Information fileのSTCInfoで定義される。STC-sequence-idは、EP_mapを持つＡＶストリームファイルで使用するものであり、TU_mapを持つＡＶストリーム

ファイルではオプションである。

プログラムは、エレメンタリストリームの集まりであり、これらのストリームの同期再生のために、ただ１つのシステムタイムベースを共有するものである。再生装置（図１の記録再生装置１）にとって、ＡＶストリームのデコードに先だち、そのＡＶストリームの内容がわかることは有用である。例えば、ビデオやオーディオのエレメンタリーストリームを伝送するトランスポートパケットのＰＩＤの値や、ビデオやオーディオのコンポーネント種類（例えば、ＨＤＴＶのビデオとＭＰＥＧ－２ ＡＡＣのオーディオストリーム等）等の情報である。この情報はＡＶストリームを参照するところのPlayListの内容をユーザに説明するところのメニュー画面を作成するのに有用であるし、また、ＡＶストリームのデコードに先だって、再生装置のＡＶデコーダ及びデマルチプレクサの初期状態をセットするために役立つ。この理由のために、Clip Information fileは、プログラムの内容を説明するためのProgramInfoを持つ。

ＭＰＥＧ２トランスポートストリームをストアしているＡＶストリームファイルは、ファイルの中でプログラム内容が変化するかもしれない。例えば、ビデオエレメンタリーストリームを伝送するところのトランスポートパケットのＰＩＤが変化したり、ビデオストリームのコンポーネント種類がＳＤＴＶからＨＤＴＶに変化する等である。

ProgramInfoは、ＡＶストリームファイルの中でのプログラム内容の変化点の情報をストアする。ＡＶストリームファイルの中で、このフォーマットで定めるところのプログラム内容が一定である区間をProgram-sequenceという。Program-sequenceは、EP_mapを持つＡＶストリームファイルで使用するものであり、TU_mapを持つＡＶストリームファイルではオプションである。

ここでは、セルフエンコードのストリームフォーマット（ＳＥＳＦ）を定義する。ＳＥＳＦは、アナログ入力信号を符号化する目的、及びディジタル入力信号（例えばＤＶ）をデコードしてからＭＰＥＧ２トランスポートストリームに符号化する場合に用いられる。

ＳＥＳＦは、ＭＰＥＧ－２トランスポートストリーム及びＡＶストリームについてのエレメンタリーストリームの符号化制限を定義する。記録再生装置１が、

ＳＥＳＦストリームをエンコードし、記録する場合、EP_mapが作成され、ディスクに記録される。

ディジタル放送のストリームは、次に示す方式のうちのいずれかが用いられて記録媒体１００に記録される。先ず、ディジタル放送のストリームをＳＥＳＦストリームにトランスコーディングする。この場合、記録されたストリームは、ＳＥＳＦに準拠しなければならない。この場合、EP_mapが作成されて、ディスクに記録されなければならない。

或いは、ディジタル放送ストリームを構成するエレメンタリーストリームを新しいエレメンタリストリームにトランスコーディングし、そのディジタル放送ストリームの規格化組織が定めるストリームフォーマットに準拠した新しいトランスポートストリームに再多重化する。この場合、EP_mapが作成されて、ディスクに記録されなければならない。

例えば、入力ストリームがＩＳＤＢ（日本のディジタルＢＳ放送の規格名称）準拠のＭＰＥＧ－２トランスポートストリームであり、それがＨＤＴＶビデオストリームとＭＰＥＧ ＡＡＣオーディオストリームを含むとする。ＨＤＴＶビデオストリームをＳＤＴＶビデオストリームにトランスコーディングし、そのＳＤＴＶビデオストリームとオリジナルのＡＡＣオーディオストリームをＴＳに再多重化する。ＳＤＴＶストリームと記録されるトランスポートストリームは、共にＩＳＤＢフォーマットに準拠しなければならない。

ディジタル放送のストリームが、記録媒体１００に記録される際の他の方式として、入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、そのときにEP_mapが作成されてディスクに記録される。

又は、入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、そのときにTU_mapが作成されてディスクに記録される。

次にディレクトリとファイルについて説明する。以下、記録再生装置１をＤＶＲ（Digital Video Recording）と適宜記述する。図１４はディスク上のディレクトリ構造の一例を示す図である。ＤＶＲのディスク上に必要なディレクトリは、

図１４に示したように、"DVR"ディレクトリを含むrootディレクトリ、"PLAYLIST"ディレクトリ、"CLIPINF"ディレクトリ、"M2TS"ディレクトリ、及び"DATA"ディレクトリを含む"DVR"ディレクトリである。rootディレクトリの下に、これら以外のディレクトリを作成されるようにしてもよいが、それらは、本例のアプリケーションフォーマットでは、無視されるとする。

"DVR"ディレクトリの下には、ＤＶＲアプリケーションフォーマットによって規定される全てのファイルとディレクトリがストアされる。"DVR"ディレクトリは、４個のディレクトリを含む。"PLAYLIST"ディレクトリの下には、Real PlayListとVirtual PlayListのデータベースファイルが置かれる。このディレクトリは、PlayListが１つもなくても存在する。

"CLIPINF"ディレクトリの下には、Clipのデータベースが置かれる。このディレクトリも、Clipが１つもなくても存在する。"M2TS"ディレクトリの下には、ＡＶストリームファイルが置かれる。このディレクトリは、ＡＶストリームファイルが１つもなくても存在する。"DATA"ディレクトリは、ディジタルＴＶ放送等のデータ放送のファイルがストアされる。

"DVR"ディレクトリは、次に示すファイルをストアする。"info.dvr"ファイルは、ＤＶＲディレクトリの下に作られ、アプリケーションレイヤの全体的な情報をストアする。ＤＶＲディレクトリの下には、ただ１つのinfo.dvrがなければならない。ファイル名は、info.dvrに固定されるとする。"menu.thmb"ファイルは、メニューサムネイル画像に関連する情報をストアする。ＤＶＲディレクトリの下には、０又は１つのメニューサムネイルがなければならない。ファイル名は、memu.thmbに固定されるとする。メニューサムネイル画像が１つもない場合、このファイルは、存在しなくてもよい。

"mark.thmb"ファイルは、マークサムネイル画像に関連する情報をストアする。ＤＶＲディレクトリの下には、０又は１つのマークサムネイルがなければならない。ファイル名は、mark.thmbに固定されるとする。メニューサムネイル画像が１つもない場合、このファイルは、存在しなくてもよい。

"PLAYLIST"ディレクトリは、２種類のPlayListファイルをストアするものであり、それらは、Real PlayListとVirtual PlayListである。"xxxxx.rpls" ファイ

ルは、１つのReal PlayListに関連する情報をストアする。それぞれのReal PlayList毎に、１つのファイルが作られる。ファイル名は、"xxxxx.rpls"である。ここで、"xxxxx"は、５個の０乃至９まで数字である。ファイル拡張子は、"rpls"でなければならないとする。

"yyyyy.vpls"ファイルは、１つのVirtual PlayListに関連する情報をストアする。それぞれのVirtual PlayList毎に、１つのファイルが作られる。ファイル名は、"yyyyy.vpls"である。ここで、"yyyyy"は、５個の０乃至９まで数字である。ファイル拡張子は、"vpls"でなければならないとする。

"CLIPINF"ディレクトリは、それぞれのＡＶストリームファイルに対応して、１つのファイルをストアする。"zzzzz.clpi" ファイルは、１つのＡＶストリームファイル(Clip AV stream file 又は Bridge-Clip AV stream file)に対応するClip Information fileである。ファイル名は、"zzzzz.clpi"であり、"zzzzz"は、５個の０乃至９までの数字である。ファイル拡張子は、"clpi"でなければならないとする。

"M2TS"ディレクトリは、ＡＶストリームのファイルをストアする。"zzzzz.m2ts"ファイルは、ＤＶＲシステムにより扱われるＡＶストリームファイルである。これは、Clip AV stream file又はBridge-Clip AV streamである。ファイル名は、"zzzzz.m2ts"であり、"zzzzz"は、５個の０乃至９までの数字である。ファイル拡張子は、"m2ts"でなければならないとする。

"DATA"ディレクトリは、データ放送から伝送されるデータをストアするものであり、データとは、例えば、XML fileやMHEGファイル等である。

次に、各ディレクトリ（ファイル）のシンタクスとセマンティクスを説明する。先ず、"info.dvr"ファイルについて説明する。図１５は、"info.dvr"ファイルのシンタクスを示す図である。"info.dvr"ファイルは、３個のオブジェクトから構成され、それらは、DVRVolume()、TableOfPlayLists()、及びMakerPrivateData()である。

図１５に示したinfo.dvrのシンタクスについて説明すると、TableOfPlayLists_Start_addressは、info.dvrファイルの先頭のバイトからの相対バイト数を単位として、TableOfPlayList()の先頭アドレスを示す。相対バイト数は０からカウン

トされる。

MakerPrivateData_Start_addressは、info.dvrファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData()の先頭アドレスを示す。相対バイト数は０からカウントされる。padding_word（パディングワード）は、info.dvrのシンタクスに従って挿入される。Ｎ１とＮ２は、０又は任意の正の整数である。それぞれのパディングワードは、任意の値をとるようにしてもよい。

DVRVolume()は、ボリューム（ディスク）の内容を記述する情報をストアする。図１６は、DVRVolume()のシンタクスを示す図である。図１６に示したDVR Volume()のシンタクスを説明すると、version_numberは、このDVRVolume()のバージョンナンバを示す４個のキャラクター文字を示す。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化される。

lengthは、このlengthフィールドの直後からDVRVolume()の最後までのDVRVolume()のバイト数を示す３２ビットの符号なし整数で表される。

ResumeVolume()は、ボリュームの中で最後に再生したReal PlayList又はVirtual PlayListのファイル名を記憶している。但し、Real PlayList又はVirtual PlayListの再生をユーザが中断したときの再生位置は、PlayListMark()において定義されるresume-markにストアされる。

図１７は、ResumeVolume()のシンタクスを示す図である。図１７に示したResumeVolume()のシンタクスを説明すると、valid_flagは、この１ビットのフラグが１にセットされている場合、resume_PlayList_nameフィールドが有効であることを示し、このフラグが０にセットされている場合、resume_PlayList_nameフィールドが無効であることを示す。

resume_PlayList_nameの１０バイトのフィールドは、リジュームされるべきReal PlayList又はVirtual PlayListのファイル名を示す。

図１６に示したDVRVolume()のシンタクスのなかの、UIAppInfoVolume は、ボリュームについてのユーザインタフェースアプリケーションのパラメータをストアする。図１８は、UIAppInfoVolumeのシンタクスを示す図であり、そのセマンティクスを説明すると、character_setの８ビットのフィールドは、Volume_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方

法は、図１９に示される値に対応する。

name_lengthの８ビットフィールドは、Volume_nameフィールドの中に示されるボリューム名のバイト長を示す。Volume_nameのフィールドは、ボリュームの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはボリュームの名称を示す。Volume_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入っていてもよい。

Volume_protect_flagは、ボリュームの中のコンテンツを、ユーザに制限することなしに見せてよいかどうかを示すフラグである。このフラグが１にセットされている場合、ユーザが正しくＰＩＮ番号（パスワード）を入力できたときだけ、そのボリュームのコンテンツを、ユーザに見せること（再生されること）が許可される。このフラグが０にセットされている場合、ユーザがＰＩＮ番号を入力しなくても、そのボリュームのコンテンツを、ユーザに見せることが許可される。

最初に、ユーザが、ディスクをプレーヤへ挿入した時点において、もしこのフラグが０にセットされているか、又は、このフラグが１にセットされていてもユーザがＰＩＮ番号を正しく入力できたならば、記録再生装置１は、そのディスクの中のPlayListの一覧を表示させる。それぞれのPlayListの再生制限は、volume_protect_flagとは無関係であり、それはUIAppInfoPlayList()の中に定義されるplayback_control_flagによって示される。

ＰＩＮは、４個の０乃至９までの数字で構成され、それぞれの数字は、ＩＳＯ／ＩＥＣ ６４６に従って符号化される。ref_thumbnail_indexのフィールドは、ボリュームに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのボリュームにはサムネイル画像が付加されており、そのサムネイル画像は、menu.thumファイルの中にストアされている。その画像は、menu.thumファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFFF である場合、そのボリュームにはサムネイル画像が付加されていないことを示す。

次に、図１５に示したinfo.dvrのシンタクス内のTableOfPlayLists()について説明する。TableOfPlayLists()は、PlayList(Real PlayListとVirtual PlayList)のファイル名をストアする。ボリュームに記録されている全てのPlayListファ

イルは、TableOfPlayList()の中に含まれる。TableOfPlayLists()は、ボリュームの中のPlayListのデフォルトの再生順序を示す。

図２０は、TableOfPlayLists()のシンタクスを示す図であり、そのシンタクスについて説明すると、TableOfPlayListsのversion_numberは、このTableOfPlayListsのバージョンナンバを示す４個のキャラクター文字を示す。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からTableOfPlayLists()の最後までのTableOfPlayLists()のバイト数を示す３２ビットの符号なしの整数である。number_of_PlayListsの１６ビットのフィールドは、PlayList_file_nameを含むfor-loopのループ回数を示す。この数字は、ボリュームに記録されているPlayListの数に等しくなければならない。PlayList_file_nameの１０バイトの数字は、PlayListのファイル名を示す。

図２１は、TableOfPlayLists()のシンタクスを他の例の構成を示す図である。図２１に示したシンタクスは、図２０に示したシンタクスに、UIAppinfoPlayList（後述）を含ませた構成とされている。このように、UIAppinfoPlayListを含ませた構成とすることで、TableOfPlayListsを読み出すだけで、メニュー画面を作成することが可能となる。ここでは、図２０に示したシンタクスを用いるとして以下の説明をする。

図１５に示したinfo.dvrのシンタクス内のMakersPrivateDataについて説明する。MakersPrivateDataは、記録再生装置１のメーカが、各社の特別なアプリケーションのために、MakersPrivateData()の中にメーカのプライベートデータを挿入できるように設けられている。各メーカのプライベートデータは、これを定義したメーカを識別するために標準化されたmaker_IDを持つ。MakersPrivateData()は、１つ以上のmaker_IDを含んでもよい。

所定のメーカが、プライベートデータを挿入したいときに、既に他のメーカのプライベートデータがMakersPrivateData()に含まれていた場合、他のメーカは、既にある古いプライベートデータを消去するのではなく、新しいプライベートデータをMakersPrivateData()の中に追加するようにする。このように、本実施例では、複数のメーカのプライベートデータが、１つのMakersPrivateData()に含まれ

ることが可能であるようにする。

図２２は、MakersPrivateDataのシンタクスを示す図である。図２２に示したMakersPrivateDataのシンタクスについて説明すると、version_numberは、このMakersPrivateData()のバージョンナンバを示す４個のキャラクター文字を示す。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からMakersPrivateData()の最後までのMakersPrivateData()のバイト数を示す３２ビットの符号なし整数を示す。

mpd_blocks_start_addressは、MakersPrivateData()の先頭のバイトからの相対バイト数を単位として、最初のmpd_block()の先頭バイトアドレスを示す。相対バイト数は０からカウントされる。number_of_maker_entriesは、MakersPrivateData()の中に含まれているメーカプライベートデータのエントリ数を与える１６ビットの符号なし整数である。MakersPrivateData()の中に、同じmaker_IDの値を持つメーカプライベートデータが２個以上存在してはならない。

mpd_block_sizeは、１０２４バイトを単位として、１つのmpd_blockの大きさを与える１６ビットの符号なし整数である。例えば、mpd_block_size=１ならば、それは１つのmpd_blockの大きさが１０２４バイトであることを示す。number_of_mpd_blocksは、MakersPrivateData()の中に含まれるmpd_blockの数を与える１６ビットの符号なし整数である。maker_IDは、そのメーカプライベートデータを作成したＤＶＲシステムの製造メーカを示す１６ビットの符号なし整数である。maker_IDに符号化される値は、このＤＶＲフォーマットのライセンサによって指定される。

maker_model_codeは、そのメーカプライベートデータを作成したＤＶＲシステムのモデルナンバコードを示す１６ビットの符号なし整数である。maker_model_codeに符号化される値は、このフォーマットのライセンスを受けた製造メーカによって設定される。start_mpd_block_numberは、そのメーカプライベートデータが開始されるmpd_blockの番号を示す１６ビットの符号なし整数である。メーカプライベートデータの先頭データは、mpd_blockの先頭にアラインされなければならない。start_mpd_block_numberは、mpd_blockのfor-loopの中の変数jに対応する。

mpd_lengthは、バイト単位でメーカプライベートデータの大きさを示す３２ビ

ットの符号なし整数である。mpd_blockは、メーカプライベートデータがストアされる領域である。MakersPrivateData()の中の全てのmpd_blockは、同じサイズでなければならない。

次に、Real PlayList fileとVirtual PlayList fileについて、換言すれば、xxxxx.rplsとyyyyy.vplsについて説明する。図２３は、xxxxx.rpls（Real PlayList）、又は、yyyyy.vpls（Virtual PlayList）のシンタクスを示す図である。xxxxx.rplsとyyyyy.vplsは、同一のシンタクス構成をもつ。xxxxx.rplsとyyyyy.vplsは、それぞれ、３個のオブジェクトから構成され、それらは、PlayList()、PlayListMark()、及びMakerPrivateData()である。

PlayListMark_Start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、PlayListMark()の先頭アドレスを示す。相対バイト数は０からカウントされる。

MakerPrivateData_Start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData()の先頭アドレスを示す。相対バイト数は０からカウントされる。

padding_word（パディングワード）は、PlayListファイルのシンタクスにしたがって挿入され、Ｎ１とＮ２は、０又は任意の正の整数である。それぞれのパディングワードは、任意の値をとるようにしてもよい。

ここで、既に、簡便に説明したが、PlayListについて更に説明する。ディスク内にある全てのReal PlayListによって、Bridge-Clip（後述）を除く全てのClipの中の再生区間が参照されていなければならない。且つ、２つ以上のReal PlayListが、それらのPlayItemで示される再生区間を同一のClipの中でオーバーラップさせてはならない。

図２４Ａから図２４Ｃを参照して更に説明すると、図２４Ａに示したように、全てのClipは、対応するReal PlayListが存在する。この規則は、図２４Ｂに示したように、編集作業が行われた後においても守られる。したがって、全てのClipは、何れかのReal PlayListを参照することにより、必ず視聴することが可能である。

図２４Ｃに示したように、Virtual PlayListの再生区間は、Real PlayListの再

生区間又はBridge-Clipの再生区間の中に含まれていなければならない。どのVirtual PlayListにも参照されないBridge-Clipがディスクの中に存在してはならない。

RealPlayListは、PlayItemのリストを含むが、SubPlayItemを含んではならない。Virtual PlayListは、PlayItemのリストを含み、PlayList()の中に示されるCPI_typeがEP_map typeであり、且つPlayList_typeが0（ビデオとオーディオを含むPlayList）である場合、Virtual PlayListは、１つのSubPlayItemを含むことができる。PlayList()では、SubPlayIteはオーディオのアフレコの目的にだけに使用される、そして、１つのVirtual PlayListが持つSubPlayItemの数は、０又は１でなければならない。

次に、PlayListについて説明する。図２５は、PlayListのシンタクスを示す図である。図２５に示したPlayListのシンタクスを説明すると、version_numberは、このPlayList()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からPlayList()の最後までのPlayList()のバイト数を示す３２ビットの符号なし整数である。PlayList_typeは、このPlayListのタイプを示す８ビットのフィールドであり、その一例を図２６に示す。

CPI_typeは、１ビットのフラグであり、PlayItem()及びSubPlayItem()によって参照されるClipのCPI_typeの値を示す。１つのPlayListによって参照される全てのClipは、それらのCPI()の中に定義されるCPI_typeの値が同じでなければならない。number_of_PlayItemsは、PlayListの中にあるPlayItemの数を示す１６ビットのフィールドである。

所定のPlayItem()に対応するPlayItem_idは、PlayItem()を含むfor-loopの中で、そのPlayItem()の現れる順番により定義される。PlayItem_idは、０から開始される。number_of_SubPlayItemsは、PlayListの中にあるSubPlayItemの数を示す１６ビットのフィールドである。この値は、０又は１である。付加的なオーディオストリームのパス（オーディオストリームパス）は、サブパスの一種である。

次に、図２５に示したPlayListのシンタクスのUIAppInfoPlayListについて説明する。UIAppInfoPlayListは、PlayListについてのユーザインタフェースアプリケ

ーションのパラメータをストアする。図２７は、UIAppInfoPlayListのシンタクスを示す図である。図２７に示したUIAppInfoPlayListのシンタクスを説明するに、character_setは、８ビットのフィールドであり、PlayList_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図１９に示したテーブルに準拠する値に対応する。

name_lengthは、８ビットフィールドであり、PlayList_nameフィールドの中に示されるPlayList名のバイト長を示す。PlayList_nameのフィールドは、PlayListの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはPlayListの名称を示す。PlayList_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入っていてもよい。

record_time_and_dateは、PlayListが記録されたときの日時をストアする５６ビットのフィールドである。このフィールドは、年／月／日／時／分／秒について、１４個の数字を４ビットのBinary Coded Decimal（ＢＣＤ）で符号化したものである。例えば、2001/12/23:01:02:03 は、"0x20011223010203"と符号化される。

durationは、PlayListの総再生時間を時間／分／秒の単位で示した２４ビットのフィールドである。このフィールドは、６個の数字を４ビットのBinary Coded Decimal（ＢＣＤ）で符号化したものである。例えば、01:45:30は、"0x014530"と符号化される。

valid_periodは、PlayListが有効である期間を示す３２ビットのフィールドである。このフィールドは、８個の数字を４ビットのBinary Coded Decimal（ＢＣＤ）で符号化したものである。例えば、記録再生装置１は、この有効期間の過ぎたPlayListを自動消去する、といったように用いられる。例えば、2001/05/07 は、"0x20010507"と符号化される。

maker_idは、そのPlayListを最後に更新したＤＶＲプレーヤ（記録再生装置１）の製造者を示す１６ビットの符号なし整数である。maker_idに符号化される値は、ＤＶＲフォーマットのライセンサによって割り当てられる。maker_codeは、そのPlayListを最後に更新したＤＶＲプレーヤのモデル番号を示す１６ビットの

符号なし整数である。maker_codeに符号化される値は、ＤＶＲフォーマットのライセンスを受けた製造者によって決められる。

playback_control_flagのフラグが１にセットされている場合、ユーザが正しくＰＩＮ番号を入力できた場合にだけ、そのPlayListは再生される。このフラグが０にセットされている場合、ユーザがＰＩＮ番号を入力しなくても、ユーザは、そのPlayListを視聴することができる。

write_protect_flagは、図２８Ａにテーブルを示すように、１にセットされている場合、write_protect_flagを除いて、そのPlayListの内容は、消去及び変更されない。このフラグが０にセットされている場合、ユーザは、そのPlayListを自由に消去及び変更できる。このフラグが１にセットされている場合、ユーザが、そのPlayListを消去、編集、又は上書きする前に、記録再生装置１はユーザに再確認するようなメッセージを表示させる。

write_protect_flagが０にセットされているReal PlayListが存在し、且つ、そのReal PlayListのClipを参照するVirtual PlayListが存在し、そのVirtual PlayListのwrite_protect_flagが１にセットされていてもよい。ユーザが、Real PlayListを消去しようとする場合、記録再生装置１は、そのReal PlayListを消去する前に、上記Virtual PlayListの存在をユーザに警告するか、又は、そのReal PlayListを"Minimize"する。

is_played_flagは、図２８Ｂに示すように、フラグが１にセットされている場合、そのPlayListは、記録されてから一度は再生されたことを示し、０にセットされている場合、そのPlayListは、記録されてから一度も再生されたことがないことを示す。

archiveは、図２８Ｃに示すように、そのPlayListがオリジナルであるか、コピーされたものであるかを示す２ビットのフィールドである。ref_thumbnail_index のフィールドは、PlayListを代表するサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのPlayListには、PlayListを代表するサムネイル画像が付加されており、そのサムネイル画像は、menu.thum ファイルの中にストアされている。その画像は、menu.thumファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、

0xFFFF である場合、そのPlayListには、PlayListを代表するサムネイル画像が付加されていない。

次に、PlayItemについて説明する。１つのPlayItem()は、基本的に次のデータを含む。Clipのファイル名を指定するためのClip_information_file_name、Clipの再生区間を特定するためのIN_timeとOUT_timeのペア、PlayList()において定義されるCPI_typeがEP_map typeである場合、IN_timeとOUT_timeが参照するところのSTC_sequence_id、及び、先行するPlayItemと現在のPlayItemとの接続の状態を示すところのconnection_conditionである。

PlayListが２つ以上のPlayItemから構成されるとき、これらのPlayItemはPlayListのグローバル時間軸上に、時間のギャップ又はオーバーラップなしに一列に並べられる。PlayList()において定義されるCPI_typeがEP_map typeであり、且つ現在のPlayItemがBridgeSequence()を持たないとき、そのPlayItemにおいて定義されるIN_timeとOUT_timeのペアは、STC_sequence_idによって指定される同じＳＴＣ連続区間上の時間を指していなければならない。このような例を図２９に示す。

図３０は、PlayList()において定義されるCPI_typeがEP_map typeであり、且つ現在のPlayItemがBridgeSequence()を持つとき、次に説明する規則が適用される場合を示している。現在のPlayItemに先行するPlayItemのIN_time（図の中でIN_time1と示されているもの)は、先行するPlayItemのSTC_sequence_idによって指定されるＳＴＣ連続区間上の時間を指している。先行するPlayItemのOUT_time（図の中でOUT_time1と示されているもの）は、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このOUT_timeは、後述する符号化制限に従っていなければならない。

現在のPlayItemのIN_time（図の中でIN_time2と示されているもの）は、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このIN_timeも、後述する符号化制限に従っていなければならない。現在のPlayItemのPlayItemのOUT_time（図の中でOUT_time2と示されているもの)は、現在のPlayItemのSTC_sequence_idによって指定されるＳＴＣ連続区間上の時間を指している。

図３１に示すように、PlayList()のCPI_typeがTU_map typeである場合、PlayItemのIN_timeとOUT_timeのペアは、同じClip ＡＶストリーム上の時間を指している。

PlayItemのシンタクスは、図３２に示すようになる。図３２に示したPlayItemのシンタクスを説明すると、Clip_Information_file_nameのフィールドは、Clip Information fileのファイル名を示す。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、Clip AV streamを示していなければならない。

STC_sequence_idは、８ビットのフィールドであり、PlayItemが参照するＳＴＣ連続区間のSTC_sequence_idを示す。PlayList()の中で指定されるCPI_typeがTU_map typeである場合、この８ビットフィールドは何も意味を持たず、０にセットされる。IN_timeは、３２ビットフィールドであり、PlayItemの再生開始時刻をストアする。IN_timeのセマンティクスは、図３３に示すように、PlayList()において定義されるCPI_typeによって異なる。

OUT_timeは、３２ビットフィールドであり、PlayItemの再生終了時刻をストアする。OUT_timeのセマンティクスは、図３４に示すように、PlayList()において定義されるCPI_typeによって異なる。

Connection_Conditionは、図３５に示したような先行するPlayItemと、現在のPlayItemとの間の接続状態を示す２ビットのフィールドである。図３６Ａ～図３６Ｄは、図３５に示したConnection_Conditionの各状態について説明する図である。

次に、BridgeSequenceInfoについて、図３７を参照して説明する。BridgeSequenceInfo()は、現在のPlayItemの付属情報であり、次に示す情報を持つ。Bridge-Clip AV streamファイルとそれに対応するClip Information fileを指定するBridge_Clip_Information_file_nameを含む。

また、先行するPlayItemが参照するClip AV stream上のソースパケットのアドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。このアドレスは、RSPN_exit_from_previous_Clipと称される。更に現在のPlayItemが参照するClip AV stream上のソースパケッ

トのアドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。このアドレスは、RSPN_enter_to_current_Clipという。

図３７において、RSPN_arrival_time_discontinuityは、the Bridge-Clip AV streamファイルの中でアライバルタイムベースの不連続点があるところのソースパケットのアドレスを示す。このアドレスは、ClipInfo()の中において定義される。

図３８は、BridgeSequenceinfoのシンタクスを示す図である。図３８に示したBridgeSequenceinfoのシンタクスを説明すると、Bridge_Clip_Information_file_nameのフィールドは、Bridge-Clip AV streamファイルに対応するClip Information fileのファイル名を示す。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、'Bridge-Clip AV stream'を示していなければならない。

RSPN_exit_from_previous_Clipの３２ビットフィールドは、先行するPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。RSPN_exit_from_previous_Clipは、ソースパケット番号を単位とする大きさであり、先行するPlayItemが参照するClip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。

RSPN_enter_to_current_Clipの３２ビットフィールドは、現在のPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。RSPN_exit_from_previous_Clipは、ソースパケット番号を単位とする大きさであり、現在のPlayItemが参照するClip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。

次に、SubPlayItemについて、図３９を参照して説明する。SubPlayItem()の使用は、PlayList()のCPI_typeがEP_map typeである場合だけに許される。ここでは、

SubPlayItemはオーディオのアフレコの目的のためだけに使用されるとする。SubPlayItem()は、次に示すデータを含む。先ず、PlayListの中のsub pathが参照するClipを指定するためのClip_information_file_ nameを含む。

また、Clipの中のsub pathの再生区間を指定するためのSubPath_IN_time と SubPath_OUT_timeを含む。更に、main pathの時間軸上でsub pathが再生開始する時刻を指定するためのsync_PlayItem_id と sync_start_PTS_of_PlayItemを含む。sub pathに参照されるオーディオのClip AV streamは、ＳＴＣ不連続点（システムタイムベースの不連続点）を含んではならない。sub pathに使われるClipのオーディオサンプルのクロックは、main pathのオーディオサンプルのクロックにロックされている。

図４０は、SubPlayItemのシンタクスを示す図である。図４０に示したSubPlayItemのシンタクスを説明すると、Clip_Information_file_nameのフィールドは、Clip Information fileのファイル名を示し、それはPlayListの中でsub pathによって使用される。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、Clip AV streamを示していなければならない。

SubPath_typeの８ビットのフィールドは、sub pathのタイプを示す。ここでは、図４１に示すように、'0x00'しか設定されておらず、他の値は、将来のために確保されている。

sync_PlayItem_idの８ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻が含まれるPlayItemのPlayItem_idを示す。所定のPlayItemに対応するPlayItem_idの値は、PlayList()において定義される（図２５参照）。

sync_start_PTS_of_PlayItemの３２ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻を示し、sync_PlayItem_idで参照されるPlayItem上のＰＴＳ(Presentaiotn Time Stamp)の上位３２ビットを示す。SubPath_IN_timeの３２ビットフィールドは、Sub pathの再生開始時刻をストアする。SubPath_IN_timeは、Sub Pathの中で最初のプレゼンテーションユニットに対応する３３ビット長のＰＴＳの上位３２ビットを示す。

SubPath_OUT_timeの３２ビットフィールドは、Sub pathの再生終了時刻をストアする。SubPath_OUT_timeは、次式によって算出されるPresenation_end_TSの値

の上位３２ビットを示す。

Presentation_end_TS = PTS_out + AU_duration

ここで、PTS_outは、SubPathの最後のプレゼンテーションユニットに対応する33ビット長のＰＴＳである。AU_durationは、SubPathの最後のプレゼンテーションユニットの９０ｋＨｚ単位の表示期間である。

次に、図２３に示したxxxxx.rplsとyyyyy.vplsのシンタクス内のPlayListMark()について説明する。PlayListについてのマーク情報は、このPlayListMarkにストアされる。図４２は、PlayListMarkのシンタクスを示す図である。図４２に示したPlayListMarkのシンタクスについて説明すると、version_numberは、このPlayListMark()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からPlayListMark()の最後までのPlayListMark()のバイト数を示す３２ビットの符号なし整数である。number_of_PlayList_marksは、PlayListMarkの中にストアされているマークの個数を示す１６ビットの符号なし整数である。number_of_PlayList_marks は、０であってもよい。mark_typeは、マークのタイプを示す８ビットのフィールドであり、図４３に示すテーブルに従って符号化される。

mark_time_stampの３２ビットフィールドは、マークが指定されたポイントを示すタイムスタンプをストアする。mark_time_stampのセマンティクスは、図４４に示すように、PlayList()において定義されるCPI_typeによって異なる。PlayItem_idは、マークが置かれているところのPlayItemを指定する８ビットのフィールドである。所定のPlayItemに対応するPlayItem_idの値は、PlayList()において定義される（図２５参照）。

character_setの８ビットのフィールドは、mark_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図１９に示した値に対応する。name_lengthの８ビットフィールドは、Mark_nameフィールドの中に示されるマーク名のバイト長を示す。mark_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはマークの名称を示す。Mark_nameフィールドの中で、

それら有効なキャラクター文字の後の値は、どのような値が設定されてもよい。

ref_thumbnail_indexのフィールドは、マークに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref_thumbnail_indexの値を用いて参照される（後述）。ref_thumbnail_indexフィールドが、0xFFFF である場合、そのマークにはサムネイル画像が付加されていないことを示す。

次に、Clip information fileについて説明する。zzzzz.clpi (Clip information fileファイル) は、図４５に示すように６個のオブジェクトから構成される。これらは、ClipInfo()、STC_Info()、ProgramInfo()、CPI()、ClipMark()、及びMakerPrivateData()である。ＡＶストリーム(Clip ＡＶストリーム又はBridge-Clip AV stream)とそれに対応するClip Informationファイルは、同じ数字列の"zzzzz"が使用される。

図４５に示したzzzzz.clpi (Clip information fileファイル) のシンタクスについて説明すると、ClipInfo_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipInfo()の先頭アドレスを示す。相対バイト数は０からカウントされる。

STC_Info_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、STC_Info()の先頭アドレスを示す。相対バイト数は０からカウントされる。ProgramInfo_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ProgramInfo()の先頭アドレスを示す。相対バイト数は０からカウントされる。CPI_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、CPI()の先頭アドレスを示す。相対バイト数は０からカウントされる。

ClipMark_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipMark()の先頭アドレスを示す。相対バイト数は０からカウントされる。MakerPrivateData_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData ()の先頭アドレ

スを示す。相対バイト数は０からカウントされる。padding_word（パディングワード）は、zzzzz.clpiファイルのシンタクスにしたがって挿入される。Ｎ１，Ｎ２，Ｎ３，Ｎ４、及びＮ５は、０又は任意の正の整数でなければならない。それぞれのパディングワードは、任意の値がとられるようにしてもよい。

次に、ClipInfoについて説明する。図４６は、ClipInfoのシンタクスを示す図である。ClipInfo()は、それに対応するＡＶストリームファイル（Clip ＡＶストリーム又はBridge-Clip ＡＶストリームファイル）の属性情報をストアする。

図４６に示したClipInfoのシンタクスについて説明すると、version_numberは、このClipInfo()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からClipInfo()の最後までのClipInfo()のバイト数を示す３２ビットの符号なし整数である。Clip_stream_typeの８ビットのフィールドは、図４７に示すように、Clip Informationファイルに対応するＡＶストリームのタイプを示す。それぞれのタイプのＡＶストリームのストリームタイプについては後述する。

offset_SPNの３２ビットのフィールドは、ＡＶストリーム（Clip ＡＶストリーム又はBridge-Clip ＡＶストリーム）ファイルの最初のソースパケットについてのソースパケット番号のオフセット値を与える。ＡＶストリームファイルが最初にディスクに記録されるとき、このoffset_SPNは０でなければならない。

図４８に示すように、ＡＶストリームファイルのはじめの部分が編集によって消去されたとき、offset_SPNは、０以外の値をとってもよい。ここでは、offset_SPNを参照する相対ソースパケット番号（相対アドレス）が、しばしば、RSPN_xxx（xxxは変形する。例．RSPN_EP_start）の形式でシンタクスの中に記述されている。相対ソースパケット番号は、ソースパケット番号を単位とする大きさであり、ＡＶストリームファイルの最初のソースパケットからoffset_SPNの値を初期値としてカウントされる。

ＡＶストリームファイルの最初のソースパケットから相対ソースパケット番号で参照されるソースパケットまでのソースパケットの数（SPN_xxx）は、次式で算出される。

SPN_xxx = RSPN_xxx - offset_SPN

図４８に、offset_SPNが４である場合の例を示す。

TS_recording_rateは、２４ビットの符号なし整数であり、この値は、ＤＶＲドライブ（書込部２２）へ又はＤＶＲドライブ（読出部２８）からのＡＶストリームの必要な入出力のビットレートを与える。record_time_and_dateは、Clipに対応するＡＶストリームが記録されたときの日時をストアする５６ビットのフィールドであり、年／月／日／時／分／秒について、１４個の数字を４ビットのBinary Coded Decimal（ＢＣＤ）で符号化したものである。例えば、2001/12/23:01:02:03は、"0x20011223010203"と符号化される。

durationは、Clipの総再生時間をアライバルタイムクロックに基づいた時間／分／秒の単位で示した２４ビットのフィールドである。このフィールドは、６個の数字を４ビットのBinary Coded Decimal（ＢＣＤ）で符号化したものである。例えば、01:45:30は、"0x014530"と符号化される。

time_controlled_flag:のフラグは、ＡＶストリームファイルの記録モードを示す。このtime_controlled_flagが１である場合、記録モードは、記録してからの時間経過に対してファイルサイズが比例するようにして記録されるモードであることを示し、次式に示す条件を満たさなければならない。

TS_average_rate*192/188*(t - start_time)$-\alpha$ <= size_clip(t)

<= TS_average_rate*192/188*(t - start_time)$+\alpha$

ここで、TS_average_rateは、ＡＶストリームファイルのトランスポートストリームの平均ビットレートをbytes/second の単位で表したものである。

また、上式において、ｔは、秒単位で表される時間を示し、start_timeは、ＡＶストリームファイルの最初のソースパケットが記録されたときの時刻であり、秒単位で表される。size_clip(t)は、 時刻ｔにおけるＡＶストリームファイルのサイズをバイト単位で表したものであり、例えば、start_timeから時刻ｔまでに１０個のソースパケットが記録された場合、size_clip(t)は10*192バイトである。αは、TS_average_rateに依存する定数である。

time_controlled_flagが0にセットされている場合、記録モードは、記録の時間経過とＡＶストリームのファイルサイズが比例するように制御していないことを

示す。例えば、これは入力トランスポートストリームをトランスペアレント記録する場合である。

TS_average_rateは、time_controlled_flagが１にセットされている場合、この２４ビットのフィールドは、上式で用いているTS_average_rateの値を示す。time_controlled_flagが０にセットされている場合、このフィールドは、何も意味を持たず、０にセットされなければならない。例えば、可変ビットレートのトランスポートストリームは、次に示す手順により符号化される。先ずトランスポートレートをTS_recording_rateの値にセットする。次に、ビデオストリームを可変ビットレートで符号化する。そして、ヌルパケットを使用しないことによって、間欠的にトランスポートパケットを符号化する。

RSPN_arrival_time_discontinuityの３２ビットフィールドは、Bridge-Clip AV streamファイル上でアライバルタイムベースの不連続が発生する場所の相対アドレスである。RSPN_arrival_time_discontinuityは、ソースパケット番号を単位とする大きさであり、Bridge-Clip AV streamファイルの最初のソースパケットからClipInfo() において定義されるoffset_SPNの値を初期値としてカウントされる。そのBridge-Clip AV streamファイルの中での絶対アドレスは、上述した

SPN_xxx = RSPN_xxx - offset_SPN

に基づいて算出される。

reserved_for_system_useの144ビットのフィールドは、システム用にリザーブされている。is_format_identifier_validのフラグが１であるとき、format_identifierのフィールドが有効であることを示す。is_original_network_ID_validのフラグが１である場合、original_network_IDのフィールドが有効であることを示す。is_transport_stream_ID_validのフラグが１である場合、transport_stream_IDのフィールドが有効であることを示す。is_servece_ID_validのフラグが１である場合、servece_IDのフィールドが有効であることを示す。

is_ country_code_validのフラグが１であるとき、country_codeのフィールドが有効であることを示す。format_identifierの３２ビットフィールドは、トランスポートストリームの中でregistration deascriotor (ISO/IEC13818-1で定義されている) が持つformat_identifierの値を示す。original_network_IDの１６ビ

ットフィールドは、トランスポートストリームの中で定義されているoriginal_network_IDの値を示す。transport_stream_IDの１６ビットフィールドは、トランスポートストリームの中で定義されているtransport_stream_IDの値を示す。

servece_IDの１６ビットフィールドは、トランスポートストリームの中で定義されているservece_IDの値を示す。country_codeの２４ビットのフィールドは、ISO3166によって定義されるカントリーコードを示す。それぞれのキャラクター文字は、ISO8859-1で符号化される。例えば、日本は"JPN"と表され、"0x4A 0x50 0x4E"と符号化される。stream_format_nameは、トランスポートストリームのストリーム定義をしているフォーマット機関の名称を示すＩＳＯ－６４６の１６個のキャラクターコードである。このフィールドの中の無効なバイトは、値'0xFF'がセットされる。

format_identifier、original_network_ID、transport_stream_ID、 servece_ID,country_code 、及びstream_format_nameは、トランスポートストリームのサービスプロバイダを示すものであり、これにより、オーディオやビデオストリームの符号化制限、ＳＩ(サービスインフォメーション)の規格やオーディオビデオストリーム以外のプライベートデータストリームのストリーム定義を認識することができる。これらの情報は、デコーダが、そのストリームをデコードできるか否か、そしてデコードできる場合にデコード開始前にデコーダシステムの初期設定を行うために用いることが可能である。

次に、STC_Infoについて説明する。ここでは、ＭＰＥＧ－２トランスポートストリームの中でＳＴＣの不連続点（システムタイムベースの不連続点）を含まない時間区間をSTC_sequenceと称し、Clipの中で、STC_sequenceは、STC_sequence_idの値によって特定される。図５０Ａ及び図５０Ｂは、連続なＳＴＣ区間について説明する図である。同じSTC_sequenceの中で同じＳＴＣの値は、決して現れない（但し、後述するように、Clipの最大時間長は制限されている）。したがって、同じSTC_sequenceの中で同じＰＴＳの値もまた、決して現れない。ＡＶストリームが、Ｎ（Ｎ＞０）個のＳＴＣ不連続点を含む場合、Clipのシステムタイムベースは、（Ｎ＋１）個のSTC_sequenceに分割される。

STC_Infoは、ＳＴＣの不連続（システムタイムベースの不連続）が発生する場

所のアドレスをストアする。図５１を参照して説明するように、RSPN_STC_startが、そのアドレスを示し、最後のSTC_sequenceを除くk番目（k>=0）のSTC_sequenceは、ｋ番目のRSPN_STC_startで参照されるソースパケットが到着した時刻から始まり、（ｋ＋１）番目のRSPN_STC_startで参照されるソースパケットが到着した時刻で終わる。最後のSTC_sequenceは、最後のRSPN_STC_startで参照されるソースパケットが到着した時刻から始まり、最後のソースパケットが到着した時刻で終了する。

図５２は、STC_Infoのシンタクスを示す図である。図５２に示したSTC_Infoのシンタクスについて説明すると、version_numberは、このSTC_Info()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からSTC_Info()の最後までのSTC_Info()のバイト数を示す３２ビットの符号なし整数である。CPI()のCPI_typeがTU_map typeを示す場合、このlengthフィールドは０をセットしてもよい。CPI()のCPI_typeがEP_map typeを示す場合、num_of_STC_sequencesは１以上の値でなければならない。

num_of_STC_sequencesの８ビットの符号なし整数は、Clipの中でのSTC_sequenceの数を示す。この値は、このフィールドに続くfor-loopのループ回数を示す。所定のSTC_sequenceに対応するSTC_sequence_idは、RSPN_STC_startを含むfor-loopの中で、そのSTC_sequenceに対応するRSPN_STC_startの現れる順番により定義されるものである。STC_sequence_idは、０から開始される。

RSPN_STC_startの３２ビットフィールドは、ＡＶストリームファイル上でSTC_sequenceが開始するアドレスを示す。RSPN_STC_startは、ＡＶストリームファイルの中でシステムタイムベースの不連続点が発生するアドレスを示す。RSPN_STC_startは、ＡＶストリームの中で新しいシステムタイムベースの最初のＰＣＲを持つソースパケットの相対アドレスとしてもよい。RSPN_STC_startは、ソースパケット番号を単位とする大きさであり、ＡＶストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、既に上述した

SPN_xxx = RSPN_xxx - offset_SPN

により算出される。

次に、図４５に示したzzzzz.clipのシンタクス内のProgramInfoについて説明する。図５３を参照しながら説明すると、ここでは、Clipの中で次の特徴をもつ時間区間をprogram_sequenceと呼ぶ。先ず、PCR_PIDの値が変わらない。次に、ビデオエレメンタリーストリームの数が変化しない。また、それぞれのビデオストリームについてのＰＩＤの値とそのVideoCodingInfoによって定義される符号化情報が変化しない。更に、オーディオエレメンタリーストリームの数が変化しない。また、それぞれのオーディオストリームについてのＰＩＤの値とそのAudioCodingInfoによって定義される符号化情報が変化しない。

program_sequenceは、同一の時刻において、ただ１つのシステムタイムベースを持つ。program_sequenceは、同一の時刻において、ただ１つのＰＭＴを持つ。ProgramInfo()は、program_sequenceが開始する場所のアドレスをストアする。RSPN_program_sequence_startが、そのアドレスを示す。

図５４は、ProgramInfoのシンタクスを示す図である。図５４に示したProgramInfoのシンタクを説明すると、version_numberは、このProgramInfo()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からProgramInfo()の最後までのProgramInfo()のバイト数を示す３２ビットの符号なし整数である。CPI()のCPI_typeがTU_map typeを示す場合、このlengthフィールドは０にセットされてもよい。CPI()のCPI_typeがEP_map typeを示す場合、number_of_programsは１以上の値でなければならない。

number_of_program_sequencesの８ビットの符号なし整数は、Clipの中でのprogram_sequenceの数を示す。この値は、このフィールドに続くfor-loopのループ回数を示す。Clipの中でprogram_sequenceが変化しない場合、number_of_program_sequencesは１をセットされなければならない。RSPN_program_sequence_startの３２ビットフィールドは、ＡＶストリームファイル上でプログラムシーケンスが開始する場所の相対アドレスである。

RSPN_program_sequence_startは、ソースパケット番号を単位とする大きさであり、ＡＶストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのＡＶストリームファイルの中での絶対アドレスは、

SPN_xxx = RSPN_xxx - offset_SPN

により算出される。シンタクスのfor-loopの中でRSPN_program_sequence_start値は、昇順に現れなければならない。

PCR_PIDの１６ビットフィールドは、そのprogram_sequenceに有効なＰＣＲフィールドを含むトランスポートパケットのＰＩＤを示す。number_of_videosの８ビットフィールドは、video_stream_PIDとVideoCodingInfo()を含むfor-loopのループ回数を示す。number_of_audiosの８ビットフィールドは、audio_stream_PIDとAudioCodingInfo()を含むfor-loopのループ回数を示す。video_stream_PIDの１６ビットフィールドは、そのprogram_sequenceに有効なビデオストリームを含むトランスポートパケットのＰＩＤを示す。このフィールドに続くVideoCodingInfo()は、そのvideo_stream_PIDで参照されるビデオストリームの内容を説明しなければならない。

audio_stream_PIDの１６ビットフィールドは、そのprogram_sequenceに有効なオーディオストリームを含むトランスポートパケットのＰＩＤを示す。このフィールドに続くAudioCodingInfo()は、そのaudio_stream_PIDで参照されるビデオストリームの内容を説明しなければならない。

なお、シンタクスのfor-loopの中でvideo_stream_PIDの値の現れる順番は、そのprogram_sequenceに有効なＰＭＴの中でビデオストリームのＰＩＤが符号化されている順番に等しくなければならない。また、シンタクスのfor-loopの中でaudio_stream_PIDの値の現れる順番は、そのprogram_sequenceに有効なＰＭＴの中でオーディオストリームのＰＩＤが符号化されている順番に等しくなければならない。

図５５は、図５４に示したPrograminfoのシンタクス内のVideoCodingInfoのシンタクスを示す図である。図５５に示したVideoCodingInfoのシンタクスを説明すると、video_formatの８ビットフィールドは、図５６に示すように、ProgramInf

o()の中のvideo_stream_PIDに対応するビデオフォーマットを示す。

frame_rateの８ビットフィールドは、図５７に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオのフレームレートを示す。display_aspect_ratioの８ビットフィールドは、図５８に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオの表示アスペクト比を示す。

図５９は、図５４に示したPrograminfoのシンタクス内のAudioCodingInfoのシンタクスを示す図である。図５９に示したAudioCodingInfoのシンタクスを説明すると、audio_codingの８ビットフィールドは、図６０に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオの符号化方法を示す。

audio_component_typeの８ビットフィールドは、図６１に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオのコンポーネントタイプを示す。sampling_frequencyの８ビットフィールドは、図６２に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオのサンプリング周波数を示す。

次に、図４５に示したzzzzz.clipのシンタクス内のＣＰＩ (Characteristic Point Information)について説明する。ＣＰＩは、ＡＶストリームの中の時間情報とそのファイルの中のアドレスとを関連づけるためにある。ＣＰＩには２つのタイプがあり、これらはEP_mapとTU_mapである。図６３に示すように、CPI()の中のCPI_typeがEP_map typeの場合、そのCPI()はEP_mapを含む。図６４に示すように、CPI()の中のCPI_typeがTU_map typeの場合、そのCPI()はTU_mapを含む。１つのＡＶストリームは、１つのEP_map又は１つのTU_mapを持つ。ＡＶストリームがＳＥＳＦトランスポートストリームの場合、それに対応するClipはEP_mapを持たなければならない。

図６５は、ＣＰＩのシンタクスを示す図である。図６５に示したＣＰＩのシンタクスを説明すると、version_numberは、このCPI()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からCPI()の最後までのCPI()のバイト数を示す３２ビットの符号なし整数である。CPI_typeは、図６６に示すように、１ビットのフラグであり、ClipのＣＰＩのタイ

プを表す。

次に、図６５に示したＣＰＩのシンタクス内のEP_mapについて説明する。EP_mapには、２つのタイプがあり、それはビデオストリーム用のEP_mapとオーディオストリーム用のEP_mapである。EP_mapの中のEP_map_typeが、EP_mapのタイプを区別する。Clipが１つ以上のビデオストリームを含む場合、ビデオストリーム用のEP_mapが使用されなければならない。Clipがビデオストリームを含まず、１つ以上のオーディオストリームを含む場合、オーディオストリーム用のEP_mapが使用されなければならない。

ビデオストリーム用のEP_mapについて図６７を参照して説明する。ビデオストリーム用のEP_mapは、stream_PID、PTS_EP_start、及び、RSPN_EP_startというデータを持つ。stream_PIDは、ビデオストリームを伝送するトランスポートパケットのＰＩＤを示す。PTS_EP_startは、ビデオストリームのシーケンスヘッダから始めるアクセスユニットのＰＴＳを示す。RSPN_EP_startは、ＡＶストリームの中でPTS_EP_startにより参照されるアクセスユニットの第１バイト目を含むソースポケットのアドレスを示す。

EP_map_for_one_stream_PID()と呼ばれるサブテーブルは、同じＰＩＤを持つトランスポートパケットによって伝送されるビデオストリーム毎に作られる。Clipの中に複数のビデオストリームが存在する場合、EP_mapは複数のEP_map_for_one_stream_PID()を含んでもよい。

オーディオストリーム用のEP_mapは、stream_PID、PTS_EP_start、及びRSPN_EP_startというデータを持つ。stream_PIDは、オーディオストリームを伝送するトランスポートパケットのＰＩＤを示す。PTS_EP_startは、オーディオストリームのアクセスユニットのＰＴＳを示す。RSPN_EP_startは、ＡＶストリームの中でPTS_EP_startで参照されるアクセスユニットの第１バイト目を含むソースポケットのアドレスを示す。

EP_map_for_one_stream_PID()と呼ばれるサブテーブルは、同じＰＩＤを持つトランスポートパケットによって伝送されるオーディオストリーム毎に作られる。Clipの中に複数のオーディオストリームが存在する場合、EP_mapは複数のEP_map_for_one_stream_PID()を含んでもよい。

EP_mapとSTC_Infoの関係を説明すると、１つのEP_map_for_one_stream_PID()は、ＳＴＣの不連続点に関係なく１つのテーブルに作られる。RSPN_EP_startの値とSTC_Info()において定義されるRSPN_STC_startの値を比較することにより、それぞれのSTC_sequenceに属するEP_mapのデータの境界が分かる（図６８を参照）。EP_mapは、同じＰＩＤで伝送される連続したストリームの範囲に対して、１つのEP_map_for_one_stream_PIDを持たねばならない。図６９に示したような場合、program#1とprogram#3は、同じビデオＰＩＤを持つが、データ範囲が連続していないので、それぞれのプログラム毎にEP_map_for_one_stream_PIDを持たねばならない。

図７０は、EP_mapのシンタクスを示す図である。図７０に示したEP_mapのシンタクスを説明すると、EP_typeは、４ビットのフィールドであり、図７１に示すように、EP_mapのエントリポイントタイプを示す。EP_typeは、このフィールドに続くデータフィールドのセマンティクスを示す。Clipが１つ以上のビデオストリームを含む場合、EP_typeは0('video')にセットされなければならない。又は、Clipがビデオストリームを含まず、１つ以上のオーディオストリームを含む場合、EP_typeは1('audio')にセットされなければならない。

number_of_stream_PIDsの１６ビットのフィールドは、EP_map()の中のnumber_of_stream_PIDsを変数にもつfor-loopのループ回数を示す。stream_PID(k)の１６ビットのフィールドは、EP_map_for_one_stream_PID(num_EP_entries(k))によって参照されるk番目のエレメンタリーストリーム（ビデオ又はオーディオストリーム）を伝送するトランスポートパケットのＰＩＤを示す。EP_typeが0 ('video')に等しい場合、そのエレメンタリストリームはビデオストリームでなければならない。また、EP_typeが1('audio')に等しい場合、そのエレメンタリストリームはオーディオストリームでなければならない。

num_EP_entries(k)の１６ビットのフィールドは、EP_map_for_one_stream_PID(num_EP_entries(k))によって参照されるnum_EP_entries(k)を示す。EP_map_for_one_stream_PID_Start_address(k): この３２ビットのフィールドは、EP_map()の中でEP_map_for_one_stream_PID(num_EP_entries(k))が始まる相対バイト位置を示す。この値は、EP_map()の第１バイト目からの大きさで示される。

padding_wordは、EP_map()のシンタクスに従って挿入されなければならない。XとYは、０又は任意の正の整数でなければならない。それぞれのパディングワードは、任意の値をとってもよい。

図７２は、EP_map_for_one_stream_PIDのシンタクスを示す図である。図７２に示したEP_map_for_one_stream_PIDのシンタクスを説明すると、PTS_EP_startの３２ビットのフィールドのセマンティクスは、EP_map()において定義されるEP_typeにより異なる。EP_typeが0 ('video')に等しい場合、このフィールドは、ビデオストリームのシーケンスヘッダで始まるアクセスユニットの３３ビット精度のＰＴＳの上位３２ビットを持つ。EP_typeが1 ('audio')に等しい場合、このフィールドは、オーディオストリームのアクセスユニットの３３ビット精度のＰＴＳの上位３２ビットを持つ。

RSPN_EP_startの３２ビットのフィールドのセマンティクスは、EP_map()において定義されるEP_typeにより異なる。EP_typeが0 ('video')に等しい場合、このフィールドは、ＡＶストリームの中でPTS_EP_startにより参照されるアクセスユニットのシーケンスヘッダの第１バイト目を含むソースポケットの相対アドレスを示す。又は、EP_typeが1 ('audio')に等しい場合、このフィールドは、ＡＶストリームの中でPTS_EP_startにより参照されるアクセスユニットのオーディオフレームの第一バイト目を含むソースポケットの相対アドレスを示す。

RSPN_EP_startは、ソースパケット番号を単位とする大きさであり、ＡＶストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのＡＶストリームファイルの中での絶対アドレスは、

SPN_xxx = RSPN_xxx - offset_SPN

により算出される。シンタクスのfor-loopの中でRSPN_EP_startの値は、昇順に現れなければならない。

次に、TU_mapについて、図７３を参照して説明する。TU_mapは、ソースパケットのアライバルタイムクロック（到着時刻ベースの時計）に基づいて、１つの時間軸を作る。その時間軸は、TU_map_time_axisと呼ばれる。TU_map_time_axisの原点は、TU_map()の中のoffset_timeによって示される。TU_map_time_axisは、o

ffset_timeから一定の単位に分割される。その単位を、time_unitという。

ＡＶストリームの中の各々のtime_unitの中で、最初の完全な形のソースパケットのＡＶストリームファイル上のアドレスが、TU_mapにストアされる。これらのアドレスを、RSPN_time_unit_startという。TU_map_time_axis上において、k (k>=0)番目のtime_unitが始まる時刻は、TU_start_time(k)と呼ばれる。この値は次式に基づいて算出される。

TU_start_time(k) = offset_time + k*time_unit_size

TU_start_time(k)は、４５ＭＨｚの精度を持つ。

図７５は、TU_mapのシンタクスを示す図である。図７５に示したTU_mapのシンタクスを説明すると、offset_timeの３２bit長のフィールドは、TU_map_time_axisに対するオフセットタイムを与える。この値は、Clipの中の最初のtime_unitに対するオフセット時刻を示す。offset_timeは、２７ＭＨｚ精度のアライバルタイムクロックから導き出される４５ｋＨｚクロックを単位とする大きさである。ＡＶストリームが新しいClipとして記録される場合、offset_timeは０にセットされなければならない。

time_unit_sizeの３２ビットフィールドは、time_unitの大きさを与えるものであり、それは２７ＭＨｚ精度のアライバルタイムクロックから導き出される４５ＭＨｚクロックを単位とする大きさである。time_unit_sizeは、１秒以下 (time_unit_size<=45000) にすることがよい。number_of_time_unit_entriesの３２ビットフィールドは、TU_map()の中にストアされているtime_unitのエントリ数を示す。

RSPN_time_unit_startの３２ビットフィールドは、ＡＶストリームの中でそれぞれのtime_unitが開始する場所の相対アドレスを示す。RSPN_time_unit_startは、ソースパケット番号を単位とする大きさであり、AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、

SPN_xxx = RSPN_xxx - offset_SPN

により算出される。シンタクスのfor-loopの中でRSPN_time_unit_startの値は、昇順に現れなければならない。（ｋ＋１）番目のtime_unitの中にソースパケット

が何もない場合、（ｋ＋１）番目のRSPN_time_unit_startは、k番目のRSPN_time_unit_startと等しくなければならない。

図４５に示したzzzzz.clipのシンタクス内のClipMarkについて説明する。ClipMarkは、クリップについてのマーク情報であり、ClipMarkの中にストアされる。このマークは、記録器（記録再生装置１）によってセットされるものであり、ユーザによってセットされるものではない。

図７５は、ClipMarkのシンタクスを示す図である。図７５に示したClipMarkのシンタクスを説明すると、version_numberは、このClipMark()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からClipMark()の最後までのClipMark()のバイト数を示す３２ビットの符号なし整数である。number_of_Clip_marksは、ClipMarkの中にストアされているマークの個数を示す１６ビットの符号なし整数であり、number_of_Clip_marks は、０であってもよい。mark_typeは、マークのタイプを示す８ビットのフィールドであり、図７６に示すテーブルに従って符号化される。

mark_time_stampは、３２ビットフィールドであり、マークが指定されたポイントを示すタイムスタンプをストアする。mark_time_stampのセマンティクスは、図７７に示すように、PlayList()の中のCPI_typeにより異なる。

STC_sequence_idは、CPI()の中のCPI_typeがEP_map typeを示す場合、この８ビットのフィールドは、マークが置かれているところのＳＴＣ連続区間のSTC_sequence_idを示す。CPI()の中のCPI_typeがTU_map typeを示す場合、この８ビットのフィールドは何も意味を持たず、０にセットされる。character_setの８ビットのフィールドは、mark_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図１９に示される値に対応する。

name_lengthの８ビットフィールドは、Mark_nameフィールドの中に示されるマーク名のバイト長を示す。mark_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはマークの名称を示す。mark_nameフィールドの中で、それら有効なキ

ャラクター文字の後の値は、どんな値が入っていてもよい。

ref_thumbnail_indexのフィールドは、マークに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFF F である場合、そのマークにはサムネイル画像が付加されていない。

MakersPrivateDataについては、図２２を参照して既に説明したので、その説明は省略する。

次に、サムネイルインフォメーション（Thumbnail Information）について説明する。サムネイル画像は、menu.thmbファイル又はmark.thmbファイルにストアされる。これらのファイルは同じシンタクス構造であり、ただ１つのThumbnail()を持つ。menu.thmbファイルは、メニューサムネイル画像、すなわちVolumeを代表する画像、及び、それぞれのPlayListを代表する画像をストアする。全てのメニューサムネイルは、ただ１つのmenu.thmbファイルにストアされる。

mark.thmbファイルは、マークサムネイル画像，すなわちマーク点を表すピクチャをストアする。全てのPlayList及びClipに対する全てのマークサムネイルは、ただ１つのmark.thmbファイルにストアされる。サムネイルは頻繁に追加、削除されるので、追加操作と部分削除の操作は容易に高速に実行できなければならない。この理由のため、Thumbnail()はブロック構造を有する。画像のデータはいくつかの部分に分割され、各部分は１つのtn_blockに格納される。１つの画像データは連続したtn_blockに格納される。tn_blockの列には、使用されていないtn_blockが存在してもよい。１つのサムネイル画像のバイト長は可変である。

図７８は、menu.thmbとmark.thmbのシンタクスを示す図であり、図７９は、図７８に示したmenu.thmbとmark.thmbのシンタクス内のThumbnailのシンタクスを示す図である。図７９に示したThumbnailのシンタクスについて説明すると、version_numberは、このThumbnail()のバージョンナンバを示す４個のキャラクター文字である。version_numberは、ＩＳＯ ６４６に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からThumbnail()の最後までのMakersPrivateData()のバイト数を示す３２ビットの符号なし整数である。tn_blocks_start_addressは、Thumbnail()の先頭のバイトからの相対バイト数を単位として、最初のtn_blockの先頭バイトアドレスを示す３２ビットの符号なし整数である。相対バイト数は０からカウントされる。number_of_thumbnailsは、Thumbnail()の中に含まれているサムネイル画像のエントリ数を与える１６ビットの符号なし整数である。

tn_block_sizeは、１０２４バイトを単位として、１つのtn_blockの大きさを与える１６ビットの符号なし整数である。例えば、tn_block_size＝１ならば、それは１つのtn_blockの大きさが１０２４バイトであることを示す。number_of_tn_blocksは、このThumbnail()中のtn_blockのエントリ数を表す１１６ビットの符号なし整数である。thumbnail_indexは、このthumbnail_indexフィールドから始まるforループ一回分のサムネイル情報で表されるサムネイル画像のインデクス番号を表す１６ビットの符号なし整数である。thumbnail_index として、0xFFFFという値を使用してはならない。thumbnail_index はUIAppInfoVolume()、UIAppInfoPlayList()、 PlayListMark()、及びClipMark()の中のref_thumbnail_indexによって参照される。

thumbnail_picture_formatは、サムネイル画像のピクチャフォーマットを表す８ビットの符号なし整数で、図８０に示すような値をとる。表中のDCFとPNGは"menu.thmb"内でのみ許される。マークサムネイルは、値"0x00" (MPEG-2 Video I-picture)をとらなければならない。

picture_data_sizeは、サムネイル画像のバイト長をバイト単位で示す３２ビットの符号なし整数である。start_tn_block_numberは、サムネイル画像のデータが始まるtn_blockのtn_block番号を表す１６ビットの符号なし整数である。サムネイル画像データの先頭は、tb_blockの先頭と一致していなければならない。tn_block番号は、０から始まり、tn_blockのfor-ループ中の変数ｋの値に関係する。

x_picture_lengthは、サムネイル画像のフレーム画枠の水平方向のピクセル数を表す１６ビットの符号なし整数である。y_picture_lengthは、サムネイル画像のフレーム画枠の垂直方向のピクセル数を表す１６ビットの符号なし整数である。

tn_blockは、 サムネイル画像がストアされる領域である。Thumbnail()の中の全てのtn_blockは、同じサイズ（固定長）であり、その大きさはtn_block_sizeによって定義される。

図８１Ａ及び図８１Ｂは、サムネイル画像データがどのようにtn_blockに格納されるかを模式的に表した図である。図８１Ａ及び図８１Ｂのように、各サムネイル画像データはtn_blockの先頭から始まり、1 tn_blockを超える大きさの場合は、連続する次のtn_blockを使用してストアされる。このようにすることにより、可変長であるピクチャデータが、固定長のデータとして管理することが可能となり、削除といった編集に対して簡便な処理により対応することができるようになる。

次に、ＡＶストリームファイルについて説明する。ＡＶストリームファイルは、"M2TS"ディレクトリ（図１４）にストアされる。ＡＶストリームファイルには、２つのタイプがあり、それらは、Clip ＡＶストリームとBridge-Clip ＡＶストリームファイルである。両方のＡＶストリーム共に、これ以降で定義されるＤＶＲ ＭＰＥＧ－２トランスポートストリームファイルの構造でなければならない。

先ず、ＤＶＲ ＭＰＥＧ－２ トランスポートストリームについて説明する。ＤＶＲ ＭＰＥＧ－２ トランスポートストリームの構造は、図８２に示すようになっている。ＡＶストリームファイルは、ＤＶＲ ＭＰＥＧ２トランスポートストリームの構造を持つ。ＤＶＲ ＭＰＥＧ２トランスポートストリームは、整数個のAligned unitから構成される。Aligned unitの大きさは、６１４４ バイト（２０４８＊３ バイト)である。Aligned unitは、ソースパケットの第１バイト目から始まる。ソースパケットは、１９２バイト長である。１つのソースパケットは、TP_extra_headerとトランスポートパケットからなる。TP_extra_headerは、４バイト長であり、またトランスポートパケットは、１８８バイト長である。

１つのAligned unitは、３２個のソースパケットからなる。ＤＶＲ ＭＰＥＧ２トランスポートストリームの中の最後のAligned unitも、また３２個のソースパケットからなる。よって、ＤＶＲ ＭＰＥＧ２トランスポートストリームは、Aligned unitの境界で終端する。ディスクに記録される入力トランスポートストリームのトランスポートパケットの数が３２の倍数でないとき、ヌルパケット（PID=

0x1FFFのトランスポートパケット）を持ったソースパケットを最後のAligned unitに使用しなければならない。ファイルシステムは、ＤＶＲ ＭＰＥＧ２トランスポートストリームに余分な情報を付加してはならない。

図８３に、ＤＶＲ ＭＰＥＧ－２トランスポートストリームのレコーダモデルを示す。図８３に示したレコーダは、レコーディングプロセスを規定するための概念上のモデルである。ＤＶＲ ＭＰＥＧ－２トランスポートストリームは、このモデルに従う。

ＭＰＥＧ－２トランスポートストリームの入力タイミングについて説明する。入力ＭＰＥＧ２トランスポートストリームは、フルトランスポートストリーム又はパーシャルトランスポートストリームである。入力されるＭＰＥＧ２トランスポートストリームは、ＩＳＯ／ＩＥＣ１３８１８－１又はＩＳＯ／ＩＥＣ１３８１８－９に従っていなければならない。ＭＰＥＧ２トランスポートストリームのi番目のバイトは、Ｔ－ＳＴＤ（ＩＳＯ／ＩＥＣ １３８１８－１で規定されるTransport stream system target decoder）とソースパケッタイザへ、時刻ｔ（ｉ）に同時に入力される。Ｒｐｋは、トランスポートパケットの入力レートの瞬時的な最大値である。

２７ＭＨｚ ＰＬＬ５２は、２７ＭＨｚクロックの周波数を発生する。２７ＭＨｚクロックの周波数は、ＭＰＥＧ－２トランスポートストリームのＰＣＲ (Program Clock Reference)の値にロックされる。arrival time clock counter５３は、２７ＭＨｚの周波数のパルスをカウントするバイナリーカウンターである。Arrival_time_clock(i)は、時刻ｔ(i)におけるArrival time clock counterのカウント値である。

source packetizer５４は、全てのトランスポートパケットにTP_extra_headerを付加し、ソースパケットを作る。Arrival_time_stampは、トランスポートパケットの第１バイト目がＴ－ＳＴＤとソースパケッタイザの両方へ到着する時刻を表す。Arrival_time_stamp(k)は、次式で示されるようにArrival_time_clock(k)のサンプル値であり、ここで、kはトランスポートパケットの第１バイト目を示す。

arrival_time_stamp(k) = arrival_time_clock(k)% 230

２つの連続して入力されるトランスポートパケットの時間間隔が、230/２７00

0000秒（約40秒）以上になる場合、その２つのトランスポートパケットのarrival_time_stampの差分は、230/２７000000秒になるようにセットされるべきである。レコーダは、そのようになる場合に備えてある。

smoothing buffer５５は、入力トランスポートストリームのビットレートをスムージングする。スムージングバッファは、オーバーフロウしてはならない。Rmaxは、スムージングバッファが空でないときのスムージングバッファからのソースパケットの出力ビットレートである。スムージングバッファが空であるとき、スムージングバッファからの出力ビットレートは０である。

次に、ＤＶＲ ＭＰＥＧ－２トランスポートストリームのレコーダモデルのパラメータについて説明する。Rmaxという値は、ＡＶストリームファイルに対応するClipInfo()において定義されるTS_recording_rateによって与えられる。この値は、次式により算出される。

Rmax = TS_recording_rate * 192/188

TS_recording_rateの値は、bytes/secondを単位とする大きさである。

入力トランスポートストリームがＳＥＳＦトランスポートストリームの場合、Rpkは、ＡＶストリームファイルに対応するClipInfo()において定義されるTS_recording_rateに等しくなければならない。入力トランスポートストリームがＳＥＳＦトランスポートストリームでない場合、この値はMPEG-2 transport streamのデスクリプター，例えばmaximum_bitrate_descriptorやpartial_transport_stream_descriptor等、において定義される値を参照してもよい。

smoothing buffer sizeは、入力トランスポートストリームがＳＥＳＦトランスポートストリームの場合、スムージングバッファの大きさは０である。入力トランスポートストリームがＳＥＳＦトランスポートストリームでない場合、スムージングバッファの大きさはMPEG-2 transport streamのデスクリプター、例えばsmoothing_buffer_descriptor、short_smoothing_buffer_descriptor、partial_transport_stream_descriptor等において定義される値を参照してもよい。

記録機（レコーダ）及び再生機（プレーヤ）は、十分なサイズのバッファを用意しなければならない。デフォールトのバッファサイズは、1536 bytes である。

次に、ＤＶＲ ＭＰＥＧ－２トランスポートストリームのプレーヤモデルについ

て説明する。図８４は、ＤＶＲ ＭＰＥＧ－２トランスポートストリームのプレーヤモデルを示す図である。これは、再生プロセスを規定するための概念上のモデルである。ＤＶＲ ＭＰＥＧ－２トランスポートストリームは、このモデルに従う。

２７ＭＨｚ X-tal６１は、２７ＭＨｚの周波数を発生する。２７ＭＨｚ周波数の誤差範囲は、+/-30 ppm (２７000000 +/- 810 Hz)でなければならない。arrival time clock counter６２は、２７ＭＨｚの周波数のパルスをカウントするバイナリーカウンターである。Arrival_time_clock(i)は、時刻ｔ(i)におけるArrival time clock counterのカウント値である。

smoothing buffer６４において、Rmaxは、スムージングバッファがフルでないときのスムージングバッファへのソースパケットの入力ビットレートである。スムージングバッファがフルであるとき、スムージングバッファへの入力ビットレートは０である。

ＭＰＥＧ－２トランスポートストリームの出力タイミングを説明すると、現在のソースパケットのarrival_time_stampがarrival_time_clock(i)のLSB ３０ビットの値と等しいとき、そのソースパケットのトランスポートパケットは、スムージングバッファから引き抜かれる。Rpkは、トランスポートパケットレートの瞬時的な

最大値である。スムージングバッファは、アンダーフロウしてはならない。

ＤＶＲ ＭＰＥＧ－２トランスポートストリームのプレーヤモデルのパラメータについては、上述したＤＶＲ ＭＰＥＧ－２トランスポートストリームのレコーダモデルのパラメータと同一である。

図８５は、Source packetのシンタクスを示す図である。transport_packet()は、ISO/IEC 13818-1で規定されるＭＰＥＧ－２トランスポートパケットである。図８５に示したSource packetのシンタクス内のTP_Extra_headerのシンタクスを図８６に示す。図８６に示したTP_Extra_headerのシンタクスについて説明すると、copy_permission_indicatorは、トランスポートパケットのペイロードのコピー制限を表す整数である。コピー制限は、copy free、no more copy、copy once、又はcopy prohibitedとすることができる。図８７は、copy_permission_indicatorの値と、それらによって指定されるモードの関係を示す。

copy_permission_indicatorは、全てのトランスポートパケットに付加される。IEEE1394ディジタルインタフェースを使用して入力トランスポートストリームを記録する場合、copy_permission_indicatorの値は、IEEE1394 isochronous packet headerの中のEMI (Encryption Mode Indicator)の値に関連付けてもよい。IEEE1394ディジタルインタフェースを使用しないで入力トランスポートストリームを記録する場合、copy_permission_indicatorの値は、トランスポートパケットの中に埋め込まれたCCIの値に関連付けてもよい。アナログ信号入力をセルフエンコードする場合、copy_permission_indicatorの値は、アナログ信号のCGMS-Aの値に関連付けてもよい。

arrival_time_stampは、次式

arrival_time_stamp(k) = arrival_time_clock(k)% 230

において、arrival_time_stampによって指定される値を持つ整数値である。

Clip ＡＶストリームの定義をするに、Clip ＡＶストリームは、上述したような定義がされるＤＶＲ ＭＰＥＧ－２トランスポートストリームの構造を持たねばならない。arrival_time_clock(i)は、Clip ＡＶストリームの中で連続して増加しなければならない。Clip ＡＶストリームの中にシステムタイムベース（ＳＴＣベース）の不連続点が存在したとしても、そのClip ＡＶストリームのarrival_time_clock(i)は、連続して増加しなければならない。

Clip ＡＶストリームの中の開始と終了の間のarrival_time_clock(i)の差分の最大値は、２６時間でなければならない。この制限は、ＭＰＥＧ２トランスポートストリームの中にシステムタイムベース（ＳＴＣベース）の不連続点が存在しない場合に、Clip ＡＶストリームの中で同じ値のＰＴＳ(Presentation Time Stamp)が決して現れないことを保証する。ＭＰＥＧ２システムズ規格は、ＰＴＳのラップアラウンド周期を233/90000秒(約26.5時間).と規定している。

Bridge-Clip ＡＶストリームの定義をするに、Bridge-Clip ＡＶストリームは、上述したような定義がされるＤＶＲ ＭＰＥＧ－２トランスポートストリームの構造を持たねばならない。Bridge-Clip ＡＶストリームは、１つのアライバルタイムベースの不連続点を含まなければならない。アライバルタイムベースの不連続点の前後のトランスポートストリームは、後述する符号化の制限に従わなければ

ならず、且つ後述するＤＶＲ－ＳＴＤに従わなければならない。

本実施例においては、編集におけるPlayItem間のビデオとオーディオのシームレス接続をサポートする。PlayItem間をシームレス接続にすることは、プレーヤ／レコーダに"データの連続供給"と"シームレスな復号処理"を保証する。"データの連続供給"とは、ファイルシステムが、デコーダにバッファのアンダーフロウを起こさせることのないように必要なビットレートでデータを供給することを保証できることである。データのリアルタイム性を保証して、データをディスクから読み出すことができるように、データが十分な大きさの連続したブロック単位でストアされるようにする。

"シームレスな復号処理"とは、プレーヤが、デコーダの再生出力にポーズやギャップを起こさせることなく、ディスクに記録されたオーディオビデオデータを表示できることである。

シームレス接続されているPlayItemが参照するＡＶストリームについて説明する。先行するPlayItemと現在のPlayItemの接続が、シームレス表示できるように保証されているかどうかは、現在のPlayItemにおいて定義されているconnection_conditionフィールドから判断することができる。PlayItem間のシームレス接続は、Bridge-Clipを使用する方法と使用しない方法がある。

図８８は、Bridge-Clipを使用する場合の先行するPlayItemと現在のPlayItemの関係を示している。図８８においては、プレーヤが読み出すストリームデータが、影を付けて示されている。図８８に示したＴＳ１は、Clip1（Clip ＡＶストリーム）の影を付けられたストリームデータとBridge-ClipのRSPN_arrival_time_discontinuityより前の影を付けられたストリームデータからなる。

ＴＳ１のClip1の影を付けられたストリームデータは、先行するPlayItemのIN_time（図８８においてIN_time1で図示されている）に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスから、RSPN_exit_from_previous_Clipで参照されるソースパケットまでのストリームデータである。ＴＳ１に含まれるBridge-ClipのRSPN_arrival_time_discontinuityより前の影を付けられたストリームデータは、Bridge-Clipの最初のソースパケットから、RSPN_arrival_time_discontinuityで参照されるソースパケットの直前のソースパケッ

トまでのストリームデータである。

また、図８８におけるＴＳ２は、Clip2（Clip ＡＶストリーム）の影を付けられたストリームデータとBridge-ClipのRSPN_arrival_time_discontinuity以後の影を付けられたストリームデータからなる。ＴＳ２に含まれるBridge-ClipのRSPN_arrival_time_discontinuity以後の影を付けられたストリームデータは、RSPN_arrival_time_discontinuityで参照されるソースパケットから、Bridge-Clipの最後のソースパケットまでのストリームデータである。ＴＳ２のClip2の影を付けられたストリームデータは、RSPN_enter_to_current_Clipで参照されるソースパケットから、現在のPlayItemのOUT_time（図８８においてOUT_time2で図示されている）に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスまでのストリームデータである。

図８９は、Bridge-Clipを使用しない場合の先行するPlayItemと現在のPlayItemの関係を示している。この場合、プレーヤが読み出すストリームデータは、影を付けて示されている。図８９におけるＴＳ１は、Clip1（Clip ＡＶストリーム)の影を付けられたストリームデータからなる。ＴＳ１のClip1の影を付けられたストリームデータは、先行するPlayItemのIN_time（図８９においてIN_time1で図示されている）に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスから始まり、Clip1の最後のソースパケットまでのデータである。

また、図８９におけるＴＳ２は、Clip2（Clip ＡＶストリーム)の影を付けられたストリームデータからなる。

ＴＳ２のClip2の影を付けられたストリームデータは、Clip2の最初のソースパケットから始まり、現在のPlayItemのOUT_time（図８９においてOUT_time2で図示されている）に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスまでのストリームデータである。

図８８と図８９において、ＴＳ１とT2は、ソースパケットの連続したストリームである。次に、ＴＳ１とＴＳ２のストリーム規定と、それらの間の接続条件について考える。先ず、シームレス接続のための符号化制限について考える。トランスポートストリームの符号化構造の制限として、先ず、ＴＳ１とＴＳ２の中に

含まれるプログラムの数は、１でなければならない。ＴＳ１とＴＳ２の中に含まれるビデオストリームの数は、１でなければならない。ＴＳ１とＴＳ２の中に含まれるオーディオストリームの数は、２以下でなければならない。ＴＳ１とＴＳ２の中に含まれるオーディオストリームの数は、等しくなければならない。ＴＳ１及び／又はＴＳ２の中に、上記以外のエレメンタリーストリーム又はプライベートストリームが含まれていてもよい。

ビデオビットストリームの制限について説明する。図９０は、ピクチャの表示順序で示すシームレス接続の例を示す図である。接続点においてビデオストリームをシームレスに表示できるためには、OUT_time1（Clip1のOUT_time）の後とIN_time2（Clip2のIN_time）の前に表示される不必要なピクチャは、接続点付近のClipの部分的なストリームを再エンコードするプロセスにより、除去されなければならない。

図９０に示したような場合において、BridgeSequenceを使用してシームレス接続を実現する例を、図９１に示す。RSPN_arrival_time_discontinuityより前のBridge-Clipのビデオストリームは、図９０のClip1のOUT_time1に対応するピクチャまでの符号化ビデオストリームからなる。そして、そのビデオストリームは先行するClip1のビデオストリームに接続され、１つの連続でＭＰＥＧ２規格に従ったエレメンタリーストリームとなるように再エンコードされている。

同様にして、RSPN_arrival_time_discontinuity以後のBridge-Clipのビデオストリームは、図９０のClip2のIN_time2に対応するピクチャ以後の符号化ビデオストリームからなる。そして、そのビデオストリームは、正しくデコード開始することができて、これに続くClip2のビデオストリームに接続され、１つの連続でＭＰＥＧ２規格に従ったエレメンタリーストリームとなるように再エンコードされている。Bridge-Clipを作るためには、一般に、数枚のピクチャは再エンコードしなければならず、それ以外のピクチャはオリジナルのClipからコピーすることができる。

図９０に示した例の場合にBridgeSequenceを使用しないでシームレス接続を実現する例を図９２に示す。Clip1のビデオストリームは、図９０のOUT_time1に対応するピクチャまでの符号化ビデオストリームからなり、それは、１つの連続で

ＭＰＥＧ２規格に従ったエレメンタリーストリームとなるように再エンコードされている。同様にして、Clip2のビデオストリームは、図９０のClip2のIN_time2に対応するピクチャ以後の符号化ビデオストリームからなり、それは、１つの連続でＭＰＥＧ２規格に従ったエレメンタリーストリームとなるように再エンコードされている。

ビデオストリームの符号化制限について説明すると、先ず、ＴＳ１とＴＳ２のビデオストリームのフレームレートは、等しくなければならない。ＴＳ１のビデオストリームは、sequence_end_codeで終端しなければならない。ＴＳ２のビデオストリームは、Sequence Header、GOP Header、そしてI-ピクチャで開始しなければならない。ＴＳ２のビデオストリームは、クローズドGOPで開始しなければならない。

ビットストリームの中で定義されるビデオプレゼンテーションユニット（フレーム又はフィールド）は、接続点を挟んで連続でなければならない。接続点において、フレーム又はフィールドのギャップがあってはならない。接続点において、トップボトムのフィールドシーケンスは連続でなければならない。３－２プルダウンを使用するエンコードの場合は、"top_field_first" 及び "repeat_first_field"フラグを書き換える必要があるかもしれない，又はフィールドギャップの発生を防ぐために局所的に再エンコードするようにしてもよい。

オーディオビットストリームの符号化制限について説明すると、ＴＳ１とＴＳ２のオーディオのサンプリング周波数は、同じでなければならない。ＴＳ１とＴＳ２のオーディオの符号化方法（例．ＭＰＥＧ1レイヤ2，ＡＣ－３，ＳＥＳＦ ＬＰＣＭ， ＡＡＣ）は、同じでなければならない。

次に、ＭＰＥＧ－２トランスポートストリームの符号化制限について説明すると、ＴＳ１のオーディオストリームの最後のオーディオフレームは、ＴＳ１の最後の表示ピクチャの表示終了時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。ＴＳ２のオーディオストリームの最初のオーディオフレームは、ＴＳ２の最初の表示ピクチャの表示開始時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。

接続点において、オーディオプレゼンテーションユニットのシーケンスにギャ

ップがあってはならない。図９３に示すように、２オーディオフレーム区間未満のオーディオプレゼンテーションユニットの長さで定義されるオーバーラップがあってもよい。ＴＳ２のエレメンタリーストリームを伝送する最初のパケットは、ビデオパケットでなければならない。接続点におけるトランスポートストリームは、後述するＤＶＲ－ＳＴＤに従わなくてはならない。

Clip及びBridge-Clipの制限について説明すると、ＴＳ１とＴＳ２は、それぞれの中にアライバルタイムベースの不連続点を含んではならない。

以下の制限は、Bridge-Clipを使用する場合にのみ適用される。ＴＳ１の最後のソースパケットとＴＳ２の最初のソースパケットの接続点においてのみ、Bridge-Clip ＡＶストリームは、ただ１つのアライバルタイムベースの不連続点を持つ。ClipInfo()において定義されるRSPN_arrival_time_discontinuityが、その不連続点のアドレスを示し、それはＴＳ２の最初のソースパケットを参照するアドレスを示さなければならない。

BridgeSequenceInfo()において定義されるRSPN_exit_from_previous_Clipによって参照されるソースパケットは、Clip1の中のどのソースパケットでもよい。それは、Aligned unitの境界である必要はない。BridgeSequenceInfo()において定義されるRSPN_enter_to_current_Clipによって参照されるソースパケットは、Clip2の中のどのソースパケットでもよい。それは、Aligned unitの境界である必要はない。

PlayItemの制限について説明すると、先行するPlayItemのOUT_time（図８８、図８９において示されるOUT_time1）は、ＴＳ１の最後のビデオプレゼンテーションユニットの表示終了時刻を示さなければならない。現在のPlayItemのIN_time（F図８８、図８９において示されるIN_time2）は、ＴＳ２の最初のビデオプレゼンテーションユニットの表示開始時刻を示さなければならない。

Bridge-Clipを使用する場合のデータアロケーションの制限について、図９４を参照して説明すると、シームレス接続は、ファイルシステムによってデータの連続供給が保証されるように作られなければならない。これは、Clip1（Clip ＡＶストリームファイル）とClip2（Clip ＡＶストリームファイル）に接続されるBridge-Clip ＡＶストリームを、データアロケーション規定を満たすように配置す

ることによって行われなければならない。

RSPN_exit_from_previous_Clip以前のClip1（Clip ＡＶストリームファイル）のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN_exit_from_previous_Clipが選択されなければならない。Bridge-Clip ＡＶストリームのデータ長は、ハーフフラグメント以上の連続領域に配置されるように、選択されなければならない。RSPN_enter_to_current_Clip以後のClip2（Clip ＡＶストリームファイル）のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN_enter_to_current_Clipが選択されなければならない。

Bridge-Clipを使用しないでシームレス接続する場合のデータアロケーションの制限について、図９５を参照して説明すると、シームレス接続は、ファイルシステムによってデータの連続供給が保証されるように作られなければならない。これは、Clip1（Clip ＡＶストリームファイル）の最後の部分とClip2（Clip ＡＶストリームファイル）の最初の部分を、データアロケーション規定を満たすように配置することによって行われなければならない。

Clip1（Clip ＡＶストリームファイル）の最後のストリーム部分が、ハーフフラグメント以上の連続領域に配置されていなければならない。Clip2（Clip ＡＶストリームファイル）の最初のストリーム部分が、ハーフフラグメント以上の連続領域に配置されていなければならない。

次に、ＤＶＲ－ＳＴＤについて説明する。ＤＶＲ－ＳＴＤは、ＤＶＲ ＭＰＥＧ２トランスポートストリームの生成及び検証の際におけるデコード処理をモデル化するための概念モデルである。また、ＤＶＲ－ＳＴＤは、上述したシームレス接続された２つのPlayItemによって参照されるＡＶストリームの生成及び検証の際におけるデコード処理をモデル化するための概念モデルでもある。

ＤＶＲ－ＳＴＤモデルを図９６に示す。図９６に示したモデルには、ＤＶＲ ＭＰＥＧ－２トランスポートストリームプレーヤモデルが構成要素として含まれている。n, TBn, MBn, EBn, TBsys, Bsys, Rxn, Rbxn, Rxsys, Dn, Dsys, On及びPn(k)の表記方法は、ＩＳＯ／ＩＥＣ１３８１８－１のＴ－ＳＴＤに定義されているものと同じである。すなわち、次の通りである。nは、エレメンタリーストリー

ムのインデクス番号である。TBnは、エレメンタリーストリームnのトランスポートバッファである。

MBnは、エレメンタリーストリームnの多重バッファである。ビデオストリームについてのみ存在する。EBnは、エレメンタリーストリームnのエレメンタリーストリームバッファである。ビデオストリームについてのみ存在する。TBsysは、復号中のプログラムのシステム情報のための入力バッファである。Bsysは、復号中のプログラムのシステム情報のためのシステムターゲットデコーダ内のメインバッファである。Rxnは、データがTBnから取り除かれる伝送レートである。Rbxnは、PESパケットペイロードがMBnから取り除かれる伝送レートである。ビデオストリームについてのみ存在する。

Rxsysは、データがTBsysから取り除かれる伝送レートである。Dnは、エレメンタリーストリームnのデコーダである。Dsysは、復号中のプログラムのシステム情報に関するデコーダである。Onは、ビデオストリームnのre-ordering bufferである。Pn(k)は、エレメンタリーストリームnのk番目のプレゼンテーションユニットである。

ＤＶＲ－ＳＴＤのデコーディングプロセスについて説明する。単一のＤＶＲ ＭＰＥＧ－２トランスポートストリームを再生している間は、トランスポートパケットをTB1, TBn又はTBsysのバッファへ入力するタイミングは、ソースパケットのarrival_time_stampにより決定される。TB1, MB1, EB1, TBn, Bn, TBsys及びBsysのバッファリング動作の規定は、ＩＳＯ／ＩＥＣ １３８１８－１に規定されているＴ－ＳＴＤと同じである。復号動作と表示動作の規定もまた、ＩＳＯ／ＩＥＣ １３８１８－１に規定されているＴ－ＳＴＤと同じである。

シームレス接続されたPlayItemを再生している間のデコーディングプロセスについて説明する。ここでは、シームレス接続されたPlayItemによって参照される２つのＡＶストリームの再生について説明をすることにし、以後の説明では、上述した（例えば、図８８に示した）ＴＳ１とＴＳ２の再生について説明する。ＴＳ１は、先行するストリームであり、ＴＳ２は、現在のストリームである。

図９７は、あるＡＶストリーム（ＴＳ１）からそれにシームレスに接続された次のＡＶストリーム（ＴＳ２）へと移るときのトランスポートパケットの入力，

復号，表示のタイミングチャートを示す。所定のＡＶストリーム（ＴＳ１）からそれにシームレスに接続された次のＡＶストリーム（ＴＳ２）へと移る間には、ＴＳ２のアライバルタイムベースの時間軸（図９７においてＡＴＣ２で示される）は、ＴＳ１のアライバルタイムベースの時間軸（図９７においてＡＴＣ１で示される）と同じでない。

また、ＴＳ２のシステムタイムベースの時間軸（図９７においてＳＴＣ２で示される）は、ＴＳ１のシステムタイムベースの時間軸（図９７においてＳＴＣ１で示される）と同じでない。ビデオの表示は、シームレスに連続していることが要求される。オーディオのプレゼンテーションユニットの表示時間にはオーバーラップがあってもよい。

ＤＶＲ－ＳＴＤ への入力タイミングについて説明する。時刻Ｔ1までの時間、すなわち、ＴＳ１の最後のビデオパケットがＤＶＲ－ＳＴＤのTB1に入力終了するまでは、ＤＶＲ－ＳＴＤのTB1、TBn 又はTBsysのバッファへの入力タイミングは、ＴＳ１のソースパケットのarrival_time_stampによって決定される。

ＴＳ１の残りのパケットは、TS_recording_rate(ＴＳ１)のビットレートでＤＶＲ－ＳＴＤのTBn又はTBsysのバッファへ入力されなければならない。ここで、TS_recording_rate(ＴＳ１)は、Clip1に対応するClipInfo()において定義されるTS_recording_rateの値である。ＴＳ１の最後のバイトがバッファへ入力する時刻は、時刻Ｔ2である。したがって、時刻Ｔ1からＴ2までの区間では、ソースパケットのarrival_time_stampは無視される。

N1をＴＳ１の最後のビデオパケットに続くＴＳ１のトランスポートパケットのバイト数とすると、時刻Ｔ1乃至Ｔ2までの時間DT1は、N1バイトがTS_recording_rate(ＴＳ１)のビットレートで入力終了するために必要な時間であり、次式により算出される。

DT1＝Ｔ2－Ｔ1＝N1 / TS_recording_rate

(ＴＳ１)時刻Ｔ1乃至Ｔ2までの間は、RXnとRXsysの値は共に、TS_recording_rate(ＴＳ１)の値に変化する。このルール以外のバッファリング動作は、Ｔ－ＳＴＤと同じである。

Ｔ2の時刻において、arrival time clock counterは、ＴＳ２の最初のソースパ

ケットのarrival_time_stampの値にリセットされる。ＤＶＲ－ＳＴＤのTB1, TBn 又はTBsysのバッファへの入力タイミングは、ＴＳ２のソースパケットのarrival_time_stampによって決定される。RXnとRXsysは共に、Ｔ－ＳＴＤにおいて定義されている値に変化する。

付加的なオーディオバッファリング及びシステムデータバッファリングについて説明すると、オーディオデコーダとシステムデコーダは、時刻Ｔ１からＴ２までの区間の入力データを処理することができるように、Ｔ－ＳＴＤで定義されるバッファ量に加えて付加的なバッファ量（約1秒分のデータ量）が必要である。

ビデオのプレゼンテーションタイミングについて説明すると、ビデオプレゼンテーションユニットの表示は、接続点を通して、ギャップなしに連続でなければならない。ここで、ＳＴＣ１は、ＴＳ１のシステムタイムベースの時間軸（図９７ではＳＴＣ１と図示されている）とし、ＳＴＣ２は、ＴＳ２のシステムタイムベースの時間軸（図９７ではＳＴＣ２と図示されている。正確には、ＳＴＣ２は、ＴＳ２の最初のＰＣＲがＴ－ＳＴＤに入力した時刻から開始する。）とする。

ＳＴＣ１とＳＴＣ２の間のオフセットは、次のように決定される。PTS1endは、ＴＳ１の最後のビデオプレゼンテーションユニットに対応するＳＴＣ１上のＰＴＳであり、PTS2startは、ＴＳ２の最初のビデオプレゼンテーションユニットに対応するＳＴＣ２上のＰＴＳであり、Tppは、ＴＳ１の最後のビデオプレゼンテーションユニットの表示期間とすると、２つのシステムタイムベースの間のオフセットSTC_deltaは、次式により算出される。

STC_delta = PTS1end + Tpp - PTS2start

オーディオのプレゼンテーションのタイミングについて説明すると、接続点において、オーディオプレゼンテーションユニットの表示タイミングのオーバーラップがあっても良く、それは０乃至２オーディオフレーム未満である（図９７に図示されている"audio overlap"を参照）。どちらのオーディオサンプルを選択するかということと、オーディオプレゼンテーションユニットの表示を接続点の後の補正されたタイムベースに再同期することは、プレーヤ側により設定されることである。

ＤＶＲ－ＳＴＤのシステムタイムクロックについて説明すると、時刻Ｔ５にお

いて、ＴＳ１の最後のオーディオプレゼンテーションユニットが表示される。システムタイムクロックは、時刻Ｔ2からＴ5の間にオーバーラップしていてもよい。この区間では、ＤＶＲ－ＳＴＤは、システムタイムクロックを古いタイムベースの値（ＳＴＣ１）と新しいタイムベースの値（ＳＴＣ２）の間で切り替える。ＳＴＣ２の値は、次式により算出される。

STC2＝STC1－STC_delta

バッファリングの連続性について説明する。STC11video_endは、ＴＳ１の最後のビデオパケットの最後のバイトがＤＶＲ－ＳＴＤのTB1へ到着するときのシステムタイムベースＳＴＣ１上のＳＴＣの値である。STC22video_startは、ＴＳ２の最初のビデオパケットの最初のバイトがＤＶＲ－ＳＴＤのTB1へ到着するときのシステムタイムベースＳＴＣ２上のＳＴＣの値である。STC21video_endは、STC11video_end の値をシステムタイムベースＳＴＣ２上の値に換算した値である。STC21video_endは、次式により算出される。

STC21video_end = STC11video_end - STC_delta

ＤＶＲ－ＳＴＤに従うために、次の２つの条件を満たすことが要求される。先ず、ＴＳ２の最初のビデオパケットのTB1への到着タイミングは、次に示す不等式を満たさなければならない。そして、次に示す不等式を満たさなければならない。

STC22video_start > STC21video_end + ΔT1

この不等式が満たされるように、Clip１及び、又は、Clip２の部分的なストリームを再エンコード及び、又は、再多重化する必要がある場合は、その必要に応じて行われる。

次に、ＳＴＣ１とＳＴＣ２を同じ時間軸上に換算したシステムタイムベースの時間軸上において、ＴＳ１からのビデオパケットの入力とそれに続くＴＳ２からのビデオパケットの入力は、ビデオバッファをオーバーフロウ及びアンダーフローさせてはならない。

このようなシンタクス、データ構造、規則に基づくことにより、記録媒体に記録されているデータの内容、再生情報等を適切に管理することができ、もって、ユーザが再生時に適切に記録媒体に記録されているデータの内容を確認したり、所望のデータを簡便に再生できるようにすることができる。

なお、以上の例は、多重化ストリームとしてＭＰＥＧ２トランスポートストリームを例にして説明しているが、これに限らず、ＭＰＥＧ２プログラムストリームや米国のＤｉｒｅｃＴＶサービス（商標）で使用されているＤＳＳトランスポートストリームについても適用することが可能である。

次に、図９８は、PlayListファイルの別の例を示す。図９８と図２３のシンタクスの大きな違いは、UIAppInfoPlayList()をストアしている場所である。図９８の例では、UIAppInfoPlayList()がPlayList()の中から外に出されているので、UIAppInfoPlayList()の将来の情報拡張が比較的容易に行えるようになる。

version_numberは、このサムネイルヘッダ情報ファイルのバージョンナンバを示す4個の数字である。

PlayList_start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、PlayList()の先頭アドレスを示す。相対バイト数は０からカウントされる。

PlayListMark_start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、PlayListMark()の先頭アドレスを示す。相対バイト数は０からカウントされる。

MakersPrivateData_start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、MakersPrivateData()の先頭アドレスを示す。相対バイト数は０からカウントされる。

図９９は、図９８のPlayListファイルの中のUIAppInfoPlayListのシンタクスを示す。PlayList_service_typeは、PlayListファイルのタイプを示す。その一例は、図２６に示されている。また、PlayList_service_typeは、ディジタルＴＶ放送のプログラムが示すサービスタイプと同じ意味を持たせてもよい。例えば、日本のディジタルＢＳ放送の場合、サービスタイプは、テレビサービス、音声サービス、及びデータ放送サービスの３種類を持つ。PlayListが使用するClip ＡＶストリームが含むプログラムのサービスタイプを代表する値をPlayList_service_typeにセットする。

PlayList_character_setは、channel_name, PlayList_name及びPlayList_detailフィールドに符号化されているキャラクター文字の符号化方法を示す。また、

これはPlayListMarkの中のmark_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。

channel_numberは、そのPlayListが記録されるとき、ユーザによって選択された放送チャンネル番号又はサービス番号を示す。複数のPlayListが１つのPlayListにコンバインされた場合は、このフィールドはそのPlayListの代表値を示す。このフィールドが0xFFFFにセットされている場合、このフィールドは何も意味を持たない。

channel_name_lengthは、channel_nameフィールドの中に示されるチャンネル名のバイト長を示す。このフィールドは、２０以下の値である。

channel_nameは、そのPlayListが記録されるとき、ユーザによって選択された放送チャンネル又はサービスの名前を示す。このフィールドの中の左からchannel_name_lengthによって示される数のバイト数が有効なキャラクター文字であり、前記名前を示す。このフィールドの中で、それら有効なキャラクター文字に続く残りのバイトは、どんな値がセットされていてもよい。複数のPlayListが１つのPlayListにコンバインされた場合は、このフィールドはそのPlayListを代表する名前を示す。

PlayList_name_lengthは、PlayList_nameフィールドの中に示されるPlayList名のバイト長を示す。

PlayList_nameは、PlayListの名前を示す。このフィールドの中の左からPlayList_name_lengthによって示される数のバイト数が有効なキャラクター文字であり、前記名前を示す。このフィールドの中で、それら有効なキャラクター文字に続く残りのバイトは、どんな値がセットされていてもよい。

PlayList_detail_lengthは、PlayList_datailフィールドの中に示されるPlayListの詳細情報のバイト長を示す。このフィールドは、１２００以下の値である。

PlayList_detailは、PlayListの詳細情報を説明するテキストを示す。このフィールドの中の左からPlayList_detail_lengthによって示される数のバイト数が有効なキャラクター文字であり、前記テキストを示す。このフィールドの中で、それら有効なキャラクター文字に続く残りのバイトは、どんな値がセットされていてもよい。

これ以外のシンタクスフィールドの意味は、図２７に示す同名のフィールドと同じである。

図１００は、図９８のPlayListファイルの中のPlayList()のシンタクスを示す。図２５の例と比べると、UIAppInfoPlayList()がなくなった点が違うだけで、これ以外は基本的に同じである。

図１０１は、SubPlayItemのシンタクスの別例を示す。図４０の例と比べると、STC_sequence_idが追加された点が大きな違いである。

STC_sequence_idは、Clip_Information_file_nameに対応するＡＶストリームファイル上の再生区間を特定するためのSubPath_IN_timeとSubPath_OUT_timeが参照するところのＳＴＣのSTC_sequence_idを示す。SubPath_IN_timeとSubPath_OUT_timeは、STC_sequence_idによって指定される同じＳＴＣ連続区間上の時間を示す。

SubPlayItemにSTC_sequence_idを追加することにより、SubPlayItemが参照するＡＶストリームファイルがＳＴＣ不連続点を持つことが許されるようになる。

これ以外のシンタクスフィールドの意味は、図４０に示す同名のフィールドと同じである。

図１０２は、Real PlayListの作成方法を説明するフローチャートを示す。図１の記録再生装置のブロック図を参照しながら説明する。

ステップＳ１１で、制御部２３はClip ＡＶストリームを記録する。

ステップＳ１２で、制御部２３はClip ＡＶストリームのEP_mapを作成可能どうかを調べる。ステップＳ１２で、Ｙｅｓの場合はステップＳ１３へ進み、EP_mapを作成する。ステップＳ１２で、Ｎｏの場合はステップＳ１４へ進み、TU_mapを作成する。

その後、ステップＳ１５で、制御部２３はPlayListのCPI_typeをセットする。

ステップＳ１６で、制御部２３は上記Clipの全ての再生可能範囲をカバーするPlayItemからなるPlayList()を作成する。CPI_typeがEP_mapタイプの場合は、時間情報をＰＴＳベースでセットする、このとき、Clipの中にＳＴＣ不連続点があり、PlayList()が２つ以上のPlayItemからなる場合は、PlayItem間のconnection_conditionもまた決定する。CPI_typeがTU_mapタイプの場合は、時間情報をアライバルタイムベースでセットする。

ステップＳ１７で、制御部２３はUIAppInfoPlayList()を作成する。

ステップＳ１８で、制御部２３はPlayListMarkを作成する。

ステップＳ１９で、制御部２３はMakersPrivateDataを作成する。

ステップＳ２０で、制御部２３はReal PlayListファイルを記録する。

このようにして、新規にClip ＡＶストリームを記録する毎に、１つのReal PlayListファイルが作られる。

図１０３は、Virtual PlayListの作成方法を説明するフローチャートである。

ステップＳ３１で、ユーザインタフェースを通して、ディスクに記録されている１つのReal PlayListが指定される。そして、そのReal PlayListの再生範囲の中から、ユーザインタフェースを通して、ＩＮ点とＯＵＴ点で示される再生区間が指定される。CPI_typeがEP_mapタイプの場合は、再生区間をＰＴＳベースでセットし、CPI_typeがTU_mapタイプの場合は、再生区間をアライバルタイムベースでセットする。

ステップＳ３２で、制御部２３はユーザによる再生範囲の指定操作が全て終了したか調べる。ユーザが上記指示した再生区間に続けて再生する区間を選ぶ場合はステップＳ３１へ戻る。ステップＳ３２でユーザによる再生範囲の指定操作が全て終了した場合は、ステップＳ３３へ進む。

ステップＳ３３で、連続して再生される２つの再生区間の間の接続状態(connection_condition)を、ユーザがユーザインタフェースを通して決定するか、又は制御部２３が決定する。

ステップＳ３４で、CPI_typeがEP_mapタイプの場合、ユーザインタフェースを通して、ユーザがサブパス(アフレコ用オーディオ)情報を指定する。ユーザがサブパスを作成しない場合はこのステップはない。

ステップＳ３５で、制御部２３はユーザが指定した再生範囲情報、及びconnection_conditionに基づいて、PlayList()を作成する。

ステップＳ３６で、制御部２３はUIAppInfoPlayList()を作成する。

ステップＳ３７で、制御部２３はPlayListMarkを作成する。

ステップＳ３８で、制御部２３はMakersPrivateDataを作成する。

ステップＳ３９で、制御部２３はVirtual PlayListファイルを記録する。

このようにして、ディスクに記録されているReal PlayListの再生範囲の中から、ユーザが見たい再生区間を選択してその再生区間をグループ化したもの毎に、１つのVirtual PlayListファイルが作られる。

図１０４はPlayListの再生方法を説明するフローチャートである。

ステップＳ５１で、制御部２３はInfo.dvr, Clip Information file, PlayList file及びサムネイルファイルの情報を取得し、ディスクに記録されているPlayListの一覧を示すＧＵＩ画面を作成し、ユーザインタフェースを通して、ＧＵＩに表示する。

ステップＳ５２で、制御部２３はそれぞれのPlayListのUIAppInfoPlayList()に基づいて、PlayListを説明する情報をＧＵＩ画面に提示する。

ステップＳ５３で、ユーザインタフェースを通して、ＧＵＩ画面上からユーザが１つのPlayListの再生を指示する。

ステップＳ５４で、制御部２３は、CPI_typeがEP_mapタイプの場合、現在のPlayItemのSTC-sequenc-idとIN_timeのＰＴＳから、IN_timeより時間的に前で最も近いエントリポイントのあるソースパケット番号を取得する。又は制御部２３は、CPI_typeがTU_mapタイプの場合、現在のPlayItemのIN_timeから、IN_timeより時間的に前で最も近いタイムユニットの開始するソースパケット番号を取得する。

ステップＳ５５で、制御部２３は上記ステップで得られたソースパケット番号からＡＶストリームのデータを読み出し、ＡＶデコーダ２７へ供給する。

ステップＳ５６で、現在のPlayItemの時間的に前のPlayItemがあった場合は、制御部２３は、前のPlayItemと現在のPlayItemとの表示の接続処理をconnection_conditionに従って行う。

ステップＳ５７で、制御部２３は、CPI_typeがEP_mapタイプの場合、ＡＶデコーダ２７は、IN_timeのＰＴＳのピクチャから表示を開始するように指示する。又は、制御部２３は、CPI_typeがEP_mapタイプの場合、ＡＶデコーダ２７は、IN_time以後のストリームのピクチャから表示を開始するように指示する。

ステップＳ５８で、制御部２３は、ＡＶデコーダ２７にＡＶストリームのデコードを続けるように指示する。

ステップＳ５９で、制御部２３は、CPI_typeがEP_mapタイプの場合、現在表示